Application Development on Cloud Environment

Thesis submitted in partial fulfillment of the requirements for the award of degree of

Master of Engineering
in
Software Engineering

By:
Simarpreet Kaur Gill
(Roll No 800831024)

Under the supervision of:
Dr. (Mrs.) Seema Bawa
Professor

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
THAPAR UNIVERSITY
PATIALA – 147004

June 2010

Certificate

I hereby certify that the work which is being presented in the thesis entitled, **"Software Engineering"**, in partial fulfillment of the requirements for the award of degree of Master of Engineering in Software Engineering submitted in Computer Science and Engineering Department of Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of *Dr.(Mrs.) Seema Bawa* and refers other researcher's works which are duly listed in the reference section.

The matter presented in this thesis has not been submitted for the award of any other degree of this or any other university.

(***Simarpreet Kaur Gill)***

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

(**Dr.(Mrs.) Seema Bawa)**
Computer Science and Engineering Department
Thapar University

Patiala

Countersigned by

(RAJESH BHATIA)
Head
Computer Science & Engineering, Department
Thapar University
Patiala

(R.K.SHARMA)
Dean (Academic Affairs)
Thapar University,
Patiala.

Acknowledgement

I wish to express my deep gratitude to Dr.(Mrs.) Seema Bawa , Professor, Computer Science and Engineering Department, for providing her uncanny guidance and support throughout the preparation of the thesis report.

I am also heartily thankful to Mr. Maninder Singh, Associate Professor, Computer Science and Engineering Department for the motivation and inspiration that triggered me for my thesis work.

I would also like to thank all the staff members, Thapar University, and all my friends especially Shashi mam, Ajay, Pinky and Jarrar who were always there at the nee of the hour and provided all the help and support, which I required for the completion of the thesis.

Last but not the least; I would like to thank God for not letting me down at the time of crises and showing me the silver lining in the dark clouds.

Simarpreet Kaur Gill
(800831024)

Abstract

Cloud computing has become a buzzword of today. Cloud computing is not a completely new concept; it has intricate connection to the established Grid Computing paradigm, and other relevant technologies such as utility computing, cluster computing, and distributed systems in general. The term “cloud” is used as a metaphor for the internet, based on how internet is depicted in computer network diagrams.

Cloud computing promises to cut operational and capital costs and, more importantly, let IT departments focus on strategic projects instead of keeping the datacenter running. Cloud computing is much more than the simple internet. Cloud computing is a construct that allows us to access applications that actually reside at a location other than our own computer or other internet-connected device; most often, this will be a distant datacenter. There are many benefits of this construct. For instance, we don’t have to install a software (eg. Microsoft Word) on each and every system in our organization. Therefore, the cost of buying so many licenses is also saved.

The beauty of cloud computing is that some other company hosts our application. This means that they handle the cost of servers, they manage the software updates, and- depending on how we craft our contract, we pay less for the service.

We won’t need to buy any equipment, which will result in lesser capital expenditure. By having someone else host the applications, we need not buy the servers nor pay for the electricity to power and cool them.

People can simply log in and use their applications wherever they are, therefore cloud computing is convenient for telecommuters and travelling remote workers.

Cloud computing is becoming increasingly relevant, as it will enable companies involved in spreading this technology to open the doors to Web3.0.

In this thesis, we have demonstrated the use of cloud by developing an application on cloud environment (Google App Engine).

Table of Contents

List of Figures

Chapter1
Introduction

This chapter describes cloud computing in detail. It explains the various related technologies like distributed computing and Grid computing, and also the benefits of cloud environment.

1.1 Background

Cloud computing has become a buzzword of today. Cloud computing is not a completely new concept; it has intricate connection to the established Grid Computing paradigm, and other relevant technologies such as utility computing, cluster computing, and distributed systems in general[1]. The term "cloud" is used as a metaphor for the internet, based on how internet is depicted in computer network diagrams [2]. There has been a computing paradigm shift over the last half century, as illustrated in figure 1.1 [3]:

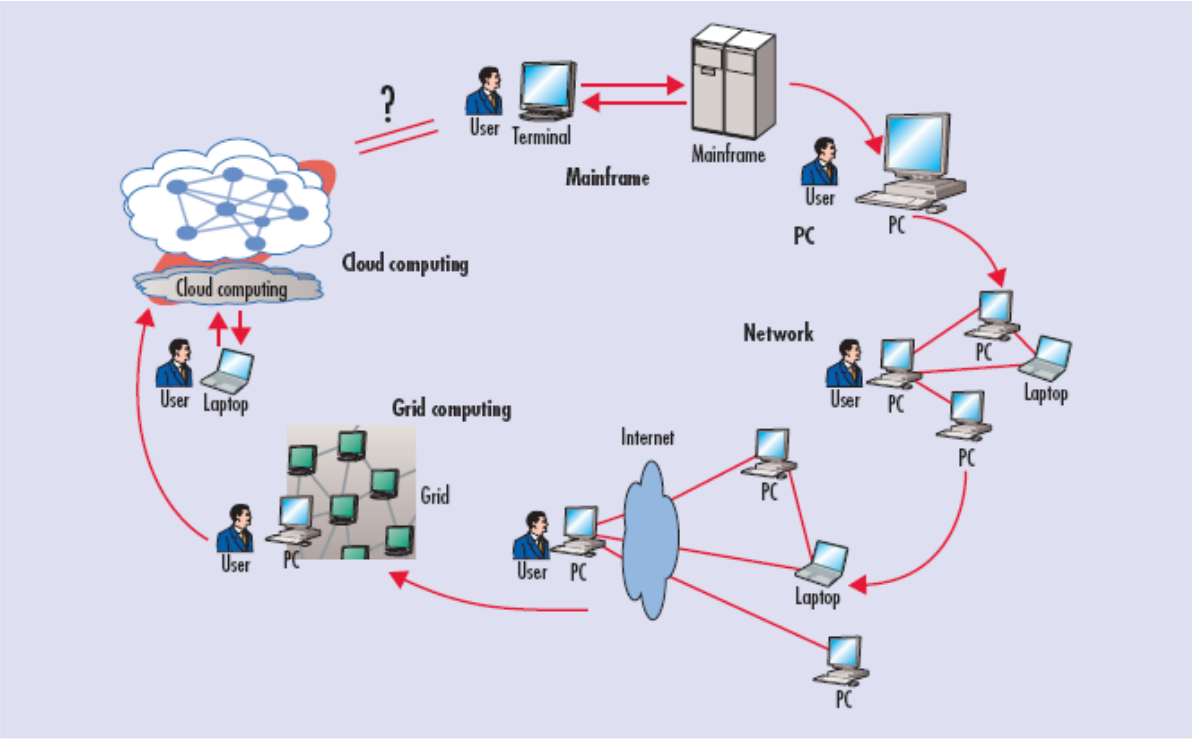

Figure 1.1: Computing paradigm shift. Over six distinct phases, computers have evolved from dummy terminals to grids and clouds [3].

In Phase1, people used terminals to connect to powerful mainframes shared by many users [3]. At that time, terminals were just little more than keyboards and monitors. In Phase2,

people started using stand-alone personal computers (PCs) which were powerful enough to satisfy users' daily requirements, and there was no need of sharing a mainframe with anyone else. In Phase3, computer networks came that allowed multiple computers to connect to each other. One could work on a PC and connect to other computers through local networks to share resources. Phase 4 was the era of advent of local networks that could connect to other local networks to establish a more global network- users could now connect to the internet to utilize remote applications and resources. Phase5 was the era of electronic Grid to facilitate shared computing power and storage resources (distributed computing). And, now in Phase6, cloud computing lets us exploit all available resources on the internet in a scalable and simple way [3]. PCs seem like lightweight terminals allowing users to utilize the cloud when compared to the infinitely powerful internet cloud. From this perspective cloud computing seems like a "return" to the original mainframe paradigm. Cloud computing has become a significant technology trend and could reshape the IT sector and the IT marketplace.

Cloud computing promises to cut operational and capital costs and, more importantly, let IT departments focus on strategic projects instead of keeping the datacenter running [4]. Cloud computing is much more than the simple internet. Cloud computing is a construct that allows us to access applications that actually reside at a location other than our own computer or other internet-connected device; most often, this will be a distant datacenter [4]. There are many benefits of this construct. For instance, we don't have to install a software (eg. Microsoft Word) on each and every system in our organization. Therefore, the cost of buying so many licenses is also saved.

The beauty of cloud computing is that some other company hosts our application. This means that they handle the cost of servers, they manage the software updates, and- depending on how we craft our contract, we pay less for the service.

We won't need to buy any equipment, which will result in lesser capital expenditure. By having someone else host the applications, we need not buy the servers nor pay for the electricity to power and cool them [4].

People can simply log in and use their applications wherever they are, therefore cloud computing is convenient for telecommuters and travelling remote workers.

Cloud computing is becoming increasingly relevant, as it will enable companies involved in spreading this technology to open the doors to Web3.0 [5].

Cloud computing promises that soon we will do our computing by renting memory and processing on a vast cloud of computers out in the network. .

The advantages for the cloud clients include the following:

- The ability to add more capacity for the peak demand
- The ability to flexibly experiment with the new services
- The ability to remove unneeded capacity when the demand slackens [6].

Therefore, we have many compelling arguments for cloud computing, but still there seems something wrong with this picture. For one thing, despite the existence of encryption and access control technologies, many organizations seem reluctant to put their precious, proprietary data out on a public cloud. Also, there is a practical and administrative problem – once all our petabytes of data are out there on the cloud, can we ever get them back?

As an individual user, a person would have a different problem- giving up ones autonomy and control.

But, the cloud advocates say that the open market will create open systems; that openness will give some service providers a competitive advantage [6].

There are scores of vendors who offer cloud services. What they have to offer varies based on the vendor and their pricing models are different.

Cloud computing is a growing field, and there will likely be new players in the market in the foreseeable future. For now, the names already known are:

*Amazon

*Google

*Microsoft

1.2 Cloud Computing

Cloud computing is a style of computing in which dynamically scalable and often virtualized resources are provided as a service over the internet. Users need not have knowledge of, expertise in, or control over the technology infrastructure in the "cloud" that supports them.

The concept generally incorporates combinations of the following:

* Infrastructure as a service (IaaS)
* Platform as a service (PaaS)
* Software as a service (SaaS)

- **Infrastructure as a Service (IaaS) :** This approach refers to the sharing of hardware resources for executing services, typically using virtualization technology.

- **Platform as a Service (PaaS) :** In this approach, the offering also includes a software execution environment, such as an application server.

- **Software as a Service (SaaS) :** SaaS is a model in which application is hosted as a service to customers who access it via the internet [4].

1.2.1 Comparison of Cloud with related technologies

There are many technologies and architectures that precede Cloud Computing in terms of attempting to "merge" computing power, share storage, and facilitate program–to-program communication. Comparison of some of these architectures is presented below.

(a) Cloud vs Web

Cloud computing is much more than simply the web. In essence, cloud computing is a construct that allows us to access applications that actually reside on some other computer [4].

The cloud operates from the idea that the work done on the client side can be moved to some unseen cluster of resources on the internet [7].

(b) Cloud computing vs Distributed Computing

Distributed computing brought us the concept of facilitating shared computing power and storage resources where as cloud computing lets us exploit all available resources on the internet in a scalable and simple way [3].

(c) Cloud computing vs Utility computing

Cloud computing is sometimes confused with utility computing. Utility computing is simply the packaging of computing resources, such as computation and storage, as a metered service similar to a traditional public utility such as electricity. Indeed, many cloud computing deployments bill like utilities- but cloud computing tends to expand what is provided by utility computing.

(d) Cloud computing vs Grid computing

Cloud computing is not a completely new concept, it has an intricate connection to Grid computing paradigm. **Cloud computing is hinting at a future in which we won't compute on local computers, but on centralized facilities operated by third party compute and storage utilities.**

In the mid 1990s, the term grid was coined to describe technologies that would allow consumers to obtain computing power on demand.

- Is "cloud computing" just a new name for the Grid?

  Yes: the vision is same-to reduce the cost of computing, increase reliability, and increase flexibility by transforming computers from something that we buy and operate ourselves to something that is operated by a third party.

  But No: things are different now than they were 10 years ago. We have a new need to analyze massive data, thus motivating greatly increased demand for computing. The prospect of needing only a credit card to get on- demand access to 1,00,000 + computers in tens of data-centers distributed throughout the world. Nevertheless, yes: the problems are mostly the same in clouds and Grids. There is a common need to be able to manage large facilities; to find methods by which consumers discover, request and use resource provided by the central facilities; and to implement the often highly parallel

computations that execute on those resources. Details differ, but the two communities are struggling with many of the same issues.

Cloud computing and grid computing function in fundamentally different ways. In grid computing, a large project is divided among multiple computers to make use of their resources. Cloud computing does just the opposite. It allows multiple smaller applications to run at the same time.

1.3 Characteristics of Clouds

- Cloud computing customers do not generally own the physical infrastructure serving as host to the software platform in question.
- Capital expenditure is reduced by renting usage from a third-party provider
- Customers consume resources as a service and pay only for resources that they use .
- Sharing "perishable and intangible" computing power can improve utilization rates, as servers are not unnecessarily left idle (which can reduce costs significantly while increasing the speed of application development).
- **Agility:** Agility improves with users able to rapidly and inexpensively re-provision technological infrastructure resources.
- **Cost:** Cost is greatly reduced and capital expenditure is converted to operational expenditure.
- **Device and location independence:** This enables users to access systems using a web browser regardless of their location or what device they are using (e.g., PC, mobile).
- **Multi-tenancy: It** enables sharing of resources and costs across a large pool of users thus allowing for:
    - **Centralization** of infrastructure in locations with lower costs (such as real estate, electricity, etc.)
    - **Peak-load capacity** increases (users need not engineer for highest possible load-levels)
    - **Utilization and efficiency** improvements for systems that are often only 10–20% utilized.
- **Reliability:** Reliability improves through the use of multiple redundant sites.
- **Scalability:** Scalability via dynamic provisioning of resources on a fine-grained, self-service basis near real-time, without users having to engineer for peak

- **Security:** Security typically improves due to centralization of data.
- **Sustainability:** Sustainability comes about through improved resource utilization, and more efficient systems

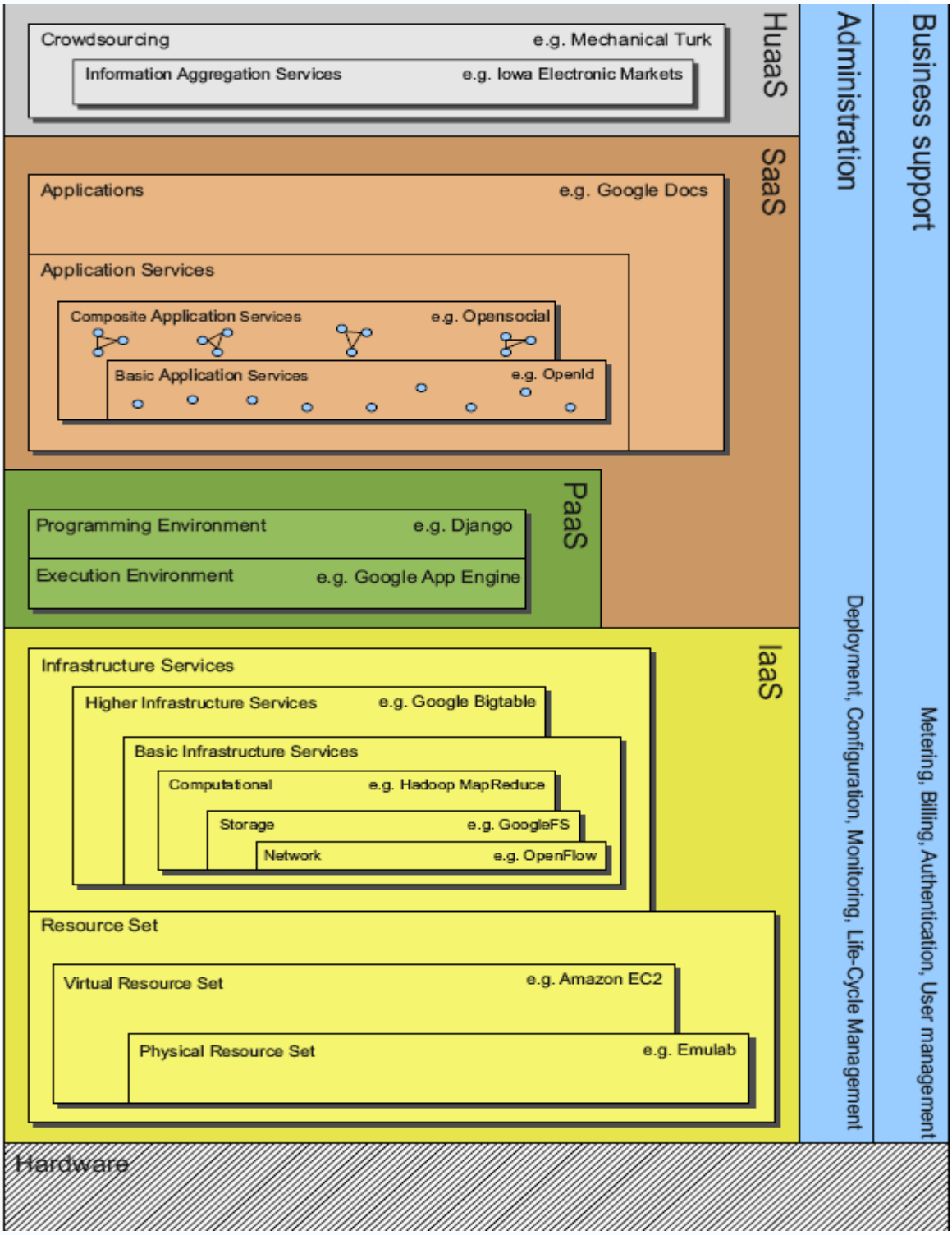

Figure 1.2: Cloud stack [8]

1.4 Types of Clouds

a) Public cloud :

Public cloud or *external cloud* describes cloud computing in the traditional mainstream sense, whereby resources are dynamically provisioned on a fine-grained, self-service basis over the Internet, via web applications/web services, from an off-site third-party provider who shares resources and bills on a fine – grained utility computing basis.

b) Hybrid cloud:

A *hybrid cloud* environment consisting of multiple internal and/or external providers will be typical for most enterprises.

c) Private cloud:

Private cloud offerings emulate cloud computing on private networks.

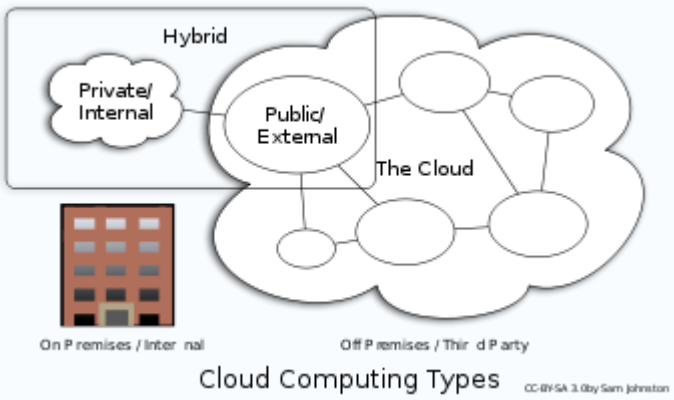

Figure 1.3: Cloud computing types [9]

1.5 Cloud Architecture

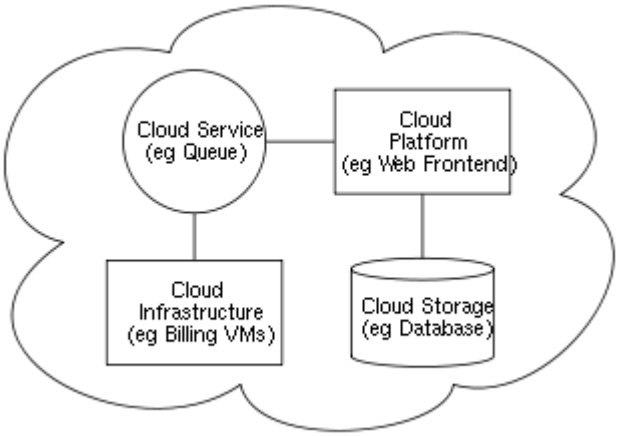

Figure1.4: Cloud computing sample architecture [7]

Cloud architecture, the systems architecture of the software systems involved in the delivery of cloud computing, comprises hardware and software designed by a cloud architect. It typically involves multiple cloud components communicating with each other over application programming interfaces, usually web services.

This closely resembles the UNIX philosophy of having multiple programs doing one thing well and working together over universal interfaces. Complexity is controlled and the resulting systems are more manageable than their monolithic counterparts [7].

Cloud architecture extends to the client, where web browsers and/or software applications access cloud applications.
Cloud storage architecture is loosely coupled, where metadata operations are centralized enabling the data nodes to scale into the hundreds, each independently delivering data to applications or users [7].

1.6 Intranets and the Cloud

By setting up thin clients to run applications and services on a local server, rather than our own desktops, we can ease the costs of deployment and maintenance, as well as reducing power costs.

Ironically, some organizations use cloud computing to deliver their corporate intranet.

1.7 Cloud Computing Open Architecture (CCOA):

Virtualization is a core technology for enabling cloud resource sharing. Service-oriented architecture (SOA) would make cloud computing platforms more flexible, extensible, and reusable.

Based on our practices in the areas of service provisioning and solution design, we think the following two key enabling technologies could play very important roles in this revolutionary phase:

a. virtualization technology
b. Service-Oriented Architecture (SOA) .

The virtualization technology handles how images of the operating systems, middleware, and applications are pro-created and allocated to the right physical machines or a slice of a server stack. On the other hand, virtualization technology can also help reuse licenses of operating systems, middleware, or software applications, once a subscriber releases his/her service from the Cloud Computing platform.

The SOA is the evolution of a system or software architecture for addressing componentization, reusability extensibility, and flexibility. In order to construct scalable Cloud Computing platforms, we need to leverage SOA to build reusable components, standard-based interfaces, and extensible solution architectures [10].

1.8 Cloud Components

In a simple, topological sense, a cloud computing solution is made up of several elements, which are as follows:

d) Clients
e) The datacenter
f) Distributed servers
g) These components make up three parts of a cloud computing solution.

Each element has a purpose and plays a particular role in delivering a functional cloud-based application [4].

h) Clients

Clients in a cloud computing architecture are exactly the same as clients in local area network (LAN). They are, typically, the computers that just sit on our desk.

Clients are the devices that the end users interact with to manage their information on the cloud. Clients generally fall into three categories:

1) **Mobile** devices include PDAs or smartphones, like a Blackberry, Wndows Mobile Smartphone, or an iPhone.
2) **Thin** Clients are computers that do not have internal hard drives, but rather let the server do all the work, but then display the information.
3) **Thick** This type of client is a regular computer, using a web browser to connect to the cloud.

i) **Datacenter**

The datacenter is the collection of servers where the application to which we subscribe is housed. It could be a large room in the basement of our building or a room full of servers on the other side of the world that we access via the internet.

A growing trend in the IT world is virtualizing servers.

j) Distributed servers

Servers are often housed at geographically disparate locations. But, to the cloud subscriber, these servers act as if they are humming away right next to each other.

1.9 Disadvantages of cloud computing

Because cloud computing does not allow users to physically possess the storage of their data (the exception being the possibility that data can be backed up to a user-owned storage device) it does leave responsibility of data storage and control in the hands of the provider.

Cloud computing has been criticized for limiting the freedom of users and making them dependent on the cloud computing provider, and some critics have alleged that is only possible to use applications or services that the provider is willing to offer.

Chapter 2

Literature Survey

This chapter describes in detail the literature survey done for the thesis work.

2.1 Evolution of Cloud Computing

Cloud computing evolved from Grid computing. This section discusses in brief about ancestors of Grid computing, then describes grid computing in detail, and finally how cloud computing evolved from and is different from Grid computing.

2.1.1 Predecessors of Grid Computing

a) Intensive Supercomputing

Intensive supercomputing is of two types which are as follows:

High Performance Computing (HPC)

- It aims at solving few "large problems" in a small amount of time.
- There are server-based solutions for problems.
- There are high performance processors.
- There is large amount of memory.
- There are fast networks.
- There is low latency time.
- There is high bandwidth.

High Throughput Computing (HTC)

- It aims at solving many "small problems" in a small amount of time.
- There are server-based solutions.
- There are large number of processors.
- There is limited size of memory.
- There are networks with medium-high latency time.

Drawbacks of this solution

- Lack of scalability
- Expensive equipments
- Expensive maintenance
- Waste of resources
- No reliability

b) Cluster Computing

A cluster is a collection of interconnected computers used as a single, unified computing resource. Computers are tightly-coupled and there is centralized job management and scheduling system.

It is based on a parallel programming model and relies on an advanced network based on Fast Ethernet or Myrinet. It maintains a good balance and cost effectiveness.

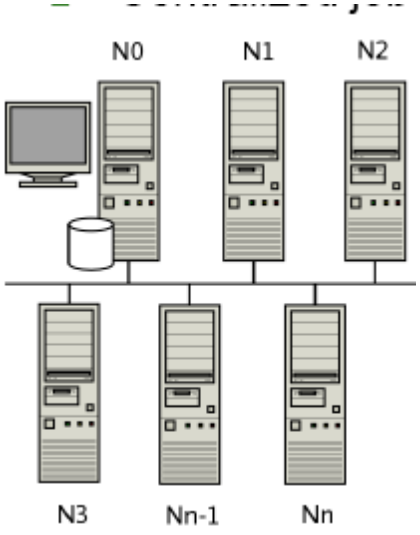

Figure 2.1: A cluster

Drawbacks of this solution

- Communications overhead
- Maintenance
- Wastage of resources

c**) Intranet Computing**

It involves use of resources of a department network and use of a management tool for running sequential or parallel jobs.

Advantages:

- More computational resources
- More low-cost CPU cycles
- Higher scalability
- More reliability
- Simplify the administration and management

Examples:

- Condor (University of Wisconsin)
- Sun Grid Engine (Sun Microsystems)
- LSF (Platform Computing)

Drawbacks of this solution

- No use of resources out of the domain of Administration
- Basic protocols and interfaces not based on open Standards
- Resource to be managed: CPU. What will happen if we need to share data?

2.1.2 Grid Computing

Grid computing, most simply stated, is distributed computing taken to the next evolutionary level. The goal is to create the illusion of a simple yet large and powerful self managing virtual computer out of a large collection of connected heterogeneous systems sharing various combinations of resources [11].

2.1.2.1What Grid Computing can do?

(a) Exploiting underutilized resources

We can use grid computing to run an existing application on a different machine when the machine on which the application is normally run might be busy due to an unusual peak in activity.

_ (b) Parallel CPU capacity

The applications can be written to use algorithms that can be partitioned into independently running parts. A CPU intensive grid application can be thought of as many smaller "subjobs," each executing on a different machine in the grid. To the extent that these subjobs do not need to communicate with each other, the more "scalable" the application becomes. A perfectly scalable application will, for example, finish10 times faster if it uses 10 times the number of processors.

Therefore, grid computing has potential for the massive parallel CPU capacity. Such computing power is driving a new evolution not only in pure sciences, but also in industries like bio-medical field, financial modelling, oil exploration, motion picture animation, and many others.

(c) Applications

Not all applications can be transformed to run in parallel on a grid and achieve scalability. There are many factors to consider in grid-enabling an application. Furthermore, there are no practical tools for transforming arbitrary applications to exploit the parallel capabilities of a grid. There are some practical tools that skilled application designers can use to write a parallel grid application. However, automatic transformation of applications is a science in its infancy.

(d)Virtual resources and virtual organizations for collaboration

Grid computing enables and simplifies collaboration among a wider audience. In the past, distributed computing promised this collaboration and achieved it to some extent. Grid computing takes these capabilities to an even wider audience, while offering important standards that enable very heterogeneous systems to work together to form the image of a large virtual computing system offering a variety of virtual resources, as illustrated in Figure 2.2 [11].

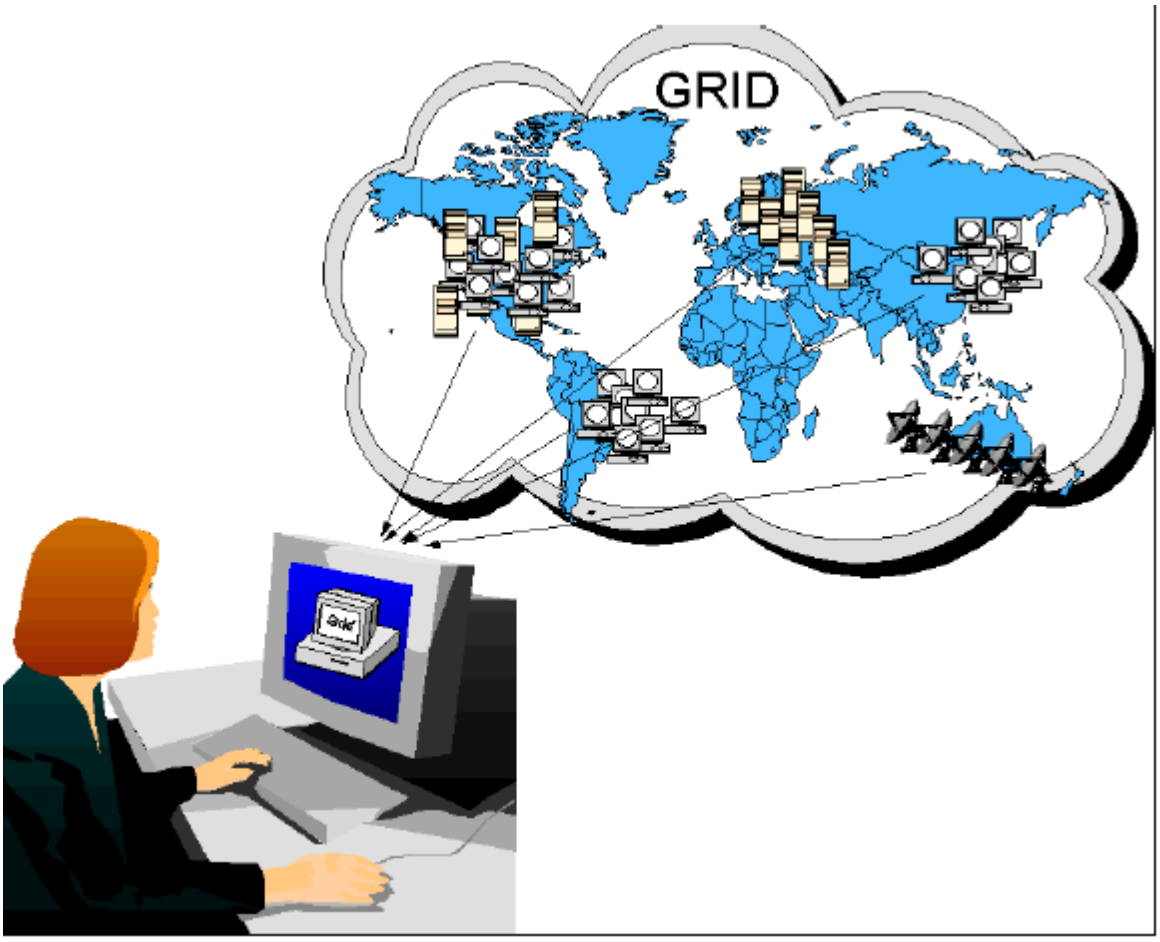

Figure2.2: The grid virtualizes heterogeneous geographically disperse resources [11]

(e) Access to additional resources

In addition to CPU and storage resources, a Grid can provide access to increased quantities of other resources and to special equipment, software, licenses, and other services [11].

(f) Resource Balancing

A Grid federates a large number of resources contributed by individual machines into a greater total virtual resource. For applications that are Grid-enabled, the Grid can offer a resource balancing effect by scheduling Grid jobs on machines with low utilization, as illustrated in Figure 2.3. This feature can be useful for handling occasional peak loads of activity in parts of larger organization. This can happen in two ways:

- An unexpected peak can be routed to relatively idle machines in the Grid.

- If the Grid is already fully utilized, the lowest priority work being performed on the Grid can be temporarily suspended or even cancelled and performed again later to make room for the higher priority work.

Without a Grid infrastructure, such balancing decisions are difficult to prioritize and execute.

(g) Reliability

In the future, we will see an approach to reliability that relies on software and hardware. A Grid is just the beginning of such technology. The systems in a Grid are relatively inexpensive and geographically dispersed. Thus, if there is a power or other kind of failure at one location, the other parts of the Grid are not likely to be affected as Grid management software can automatically resubmit jobs to other machines on the Grid when a failure is detected. In critical, real-time situations, multiple copies of the important jobs can be run on different machines throughout the grid, as illustrated in Figure 2.4 [11].

Such grid systems will utilize "autonomic computing." This is a type of software that automatically heals problems in the grid, perhaps even before an operator is aware of them. In principle, most of the reliability attributes achieved using hardware in today's high availability systems can be achieved using software in a Grid setting in the future [11].

(h) Management

The goal to virtualize the resources on the Grid and more uniformly handle heterogeneous systems will create new opportunities to better manage a larger, more disperse IT infrastructure [11]. IT departments will be able to control expenditures for computing resources over a larger organization as Grid makes it easier to visualize capacity and utilization.

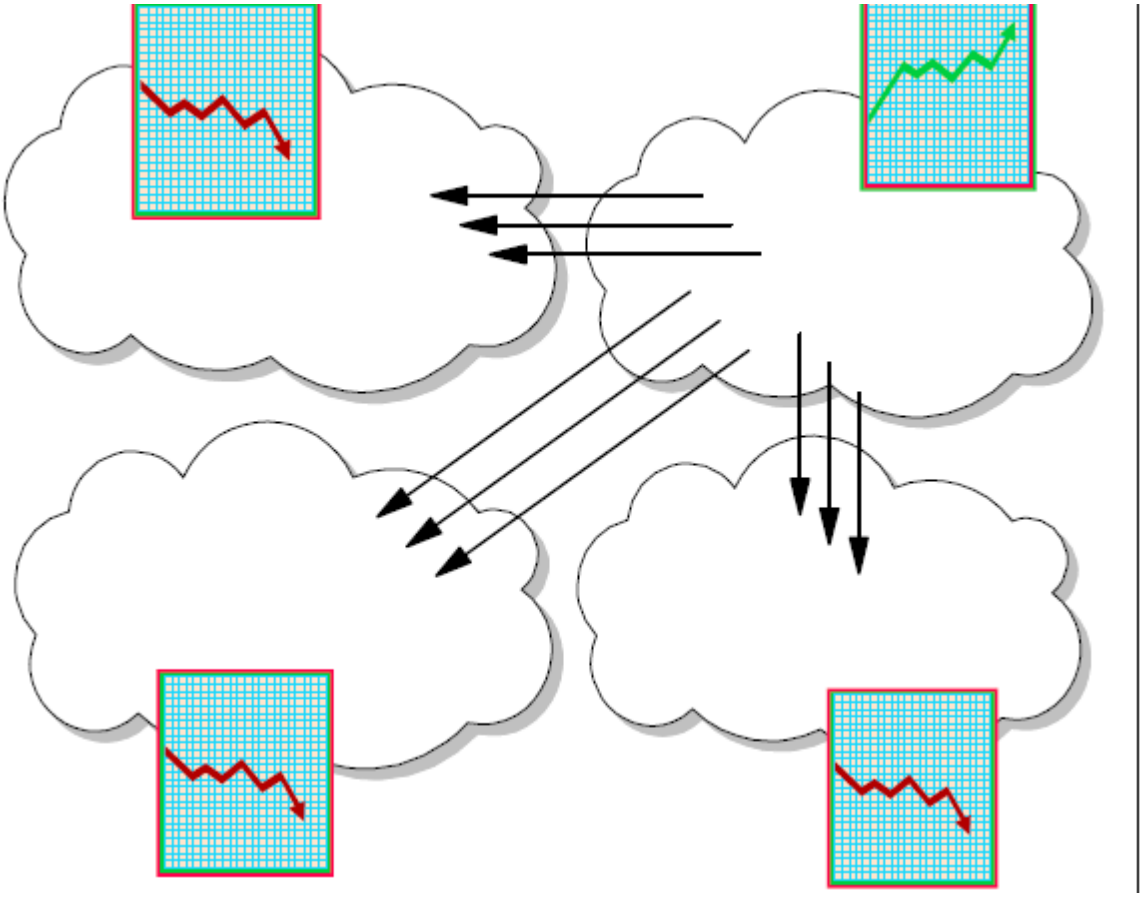

Figure2.3: jobs are migrated to less busy parts of the grid to balance loads [11]

By using the Grid, we can manage priorities among different projects. In the past, each project was responsible for its own IT resource hardware and the expenses associated with it. Often this hardware might be underutilized, while some other project might find itself in trouble, needing more resources due to unexpected events. With Grid computing, it becomes easier to control and manage such situations. As illustrated in Figure 2.5 administrators can change any number of policies that affect how the different organizations might share or compete for resources.

Aggregating utilization data over a larger set of projects can enhance an organization's ability to project future upgrade needs. When maintenance is required, Grid work can be rerouted to other machines without crippling the projects involved.

Autonomic computing can come into play here too. Various tools may be able to identify important trends throughout the Grid, informing management of those that require attention.

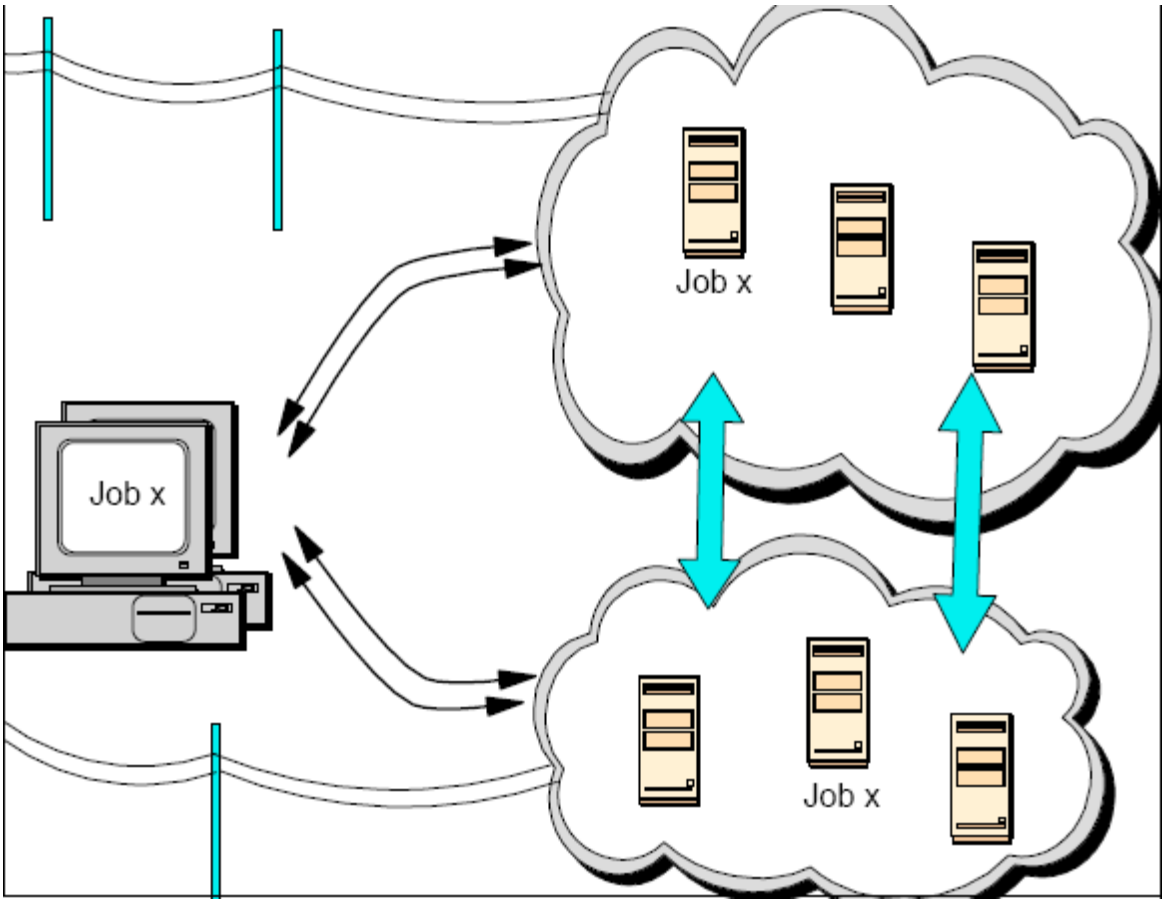

Figure2.4: Redundant Grid configuration [11]

2.1.2.1 Grid Concepts and Components

In this sub-section, we introduce the various Grid concepts, components, and terms in more detail.

A) Types of resources

A Grid is a collection of machines, sometimes referred to as "nodes," "resources," "members," "donors," "clients," "hosts," "engines," and many other such terms.

They all contribute any combination of resources to the Grid as a whole.

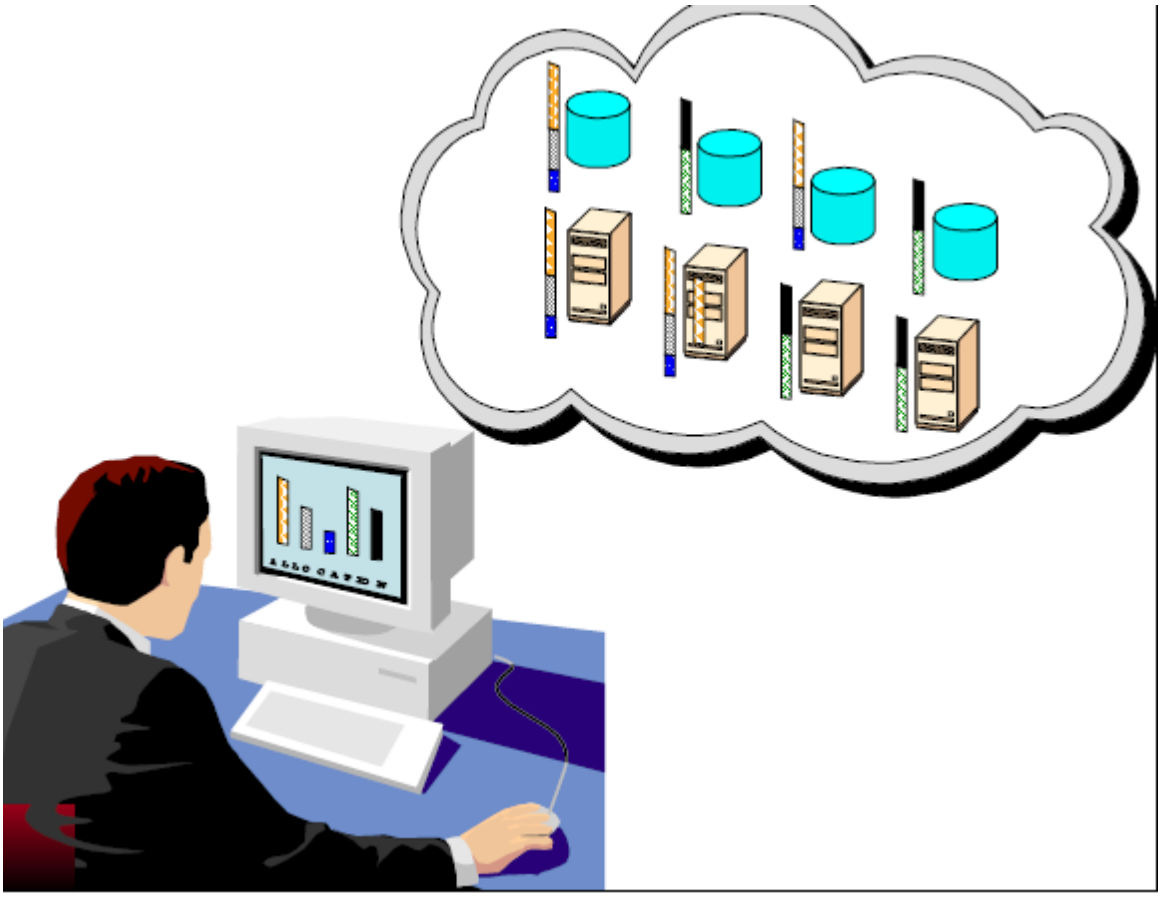

Figure 2.5: Administrators can adjust policies to better allocate resources [11]

Computation

The most common resource is computing cycles provided by the processors of the machines on the Grid. The processors can vary in speed, architecture, software platform, and other associated factors, such as memory, storage, and connectivity [11].

There are three main ways to exploit the computational resources of a Grid, which are as follows:

- To use the Grid to run an existing application on an available machine on the Grid rather than locally
- To use an application designed to split its work in such a way that the separate parts can execute in parallel on different processors
- To run an application, that needs to be executed many times, on many different machines in the Grid

Storage

The second most common resource used in a Grid is data storage. A Grid providing an integrated view of data storage is sometimes called a "Data Grid." Each machine on the Grid usually provides some quantity of storage for Grid use, even if temporary.

Capacity can be increased by using the storage on multiple machines with a unifying file system. Any individual file or database can span several storage devices and machines, eliminating maximum size restrictions often imposed by file systems shipped with operating systems.

More advanced file systems on a grid can automatically duplicate sets of data, to provide redundancy for increased reliability and increased performance. Data striping can also be implemented by grid file systems, as illustrated in Figure 2.6 [11].

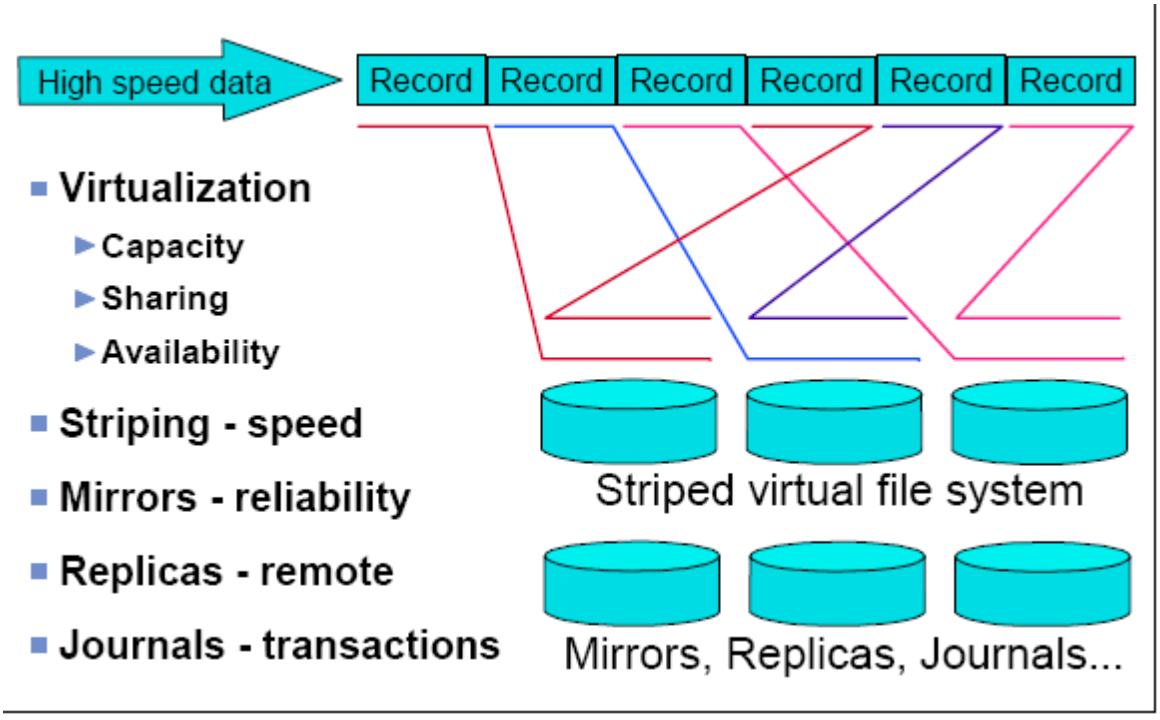

Figure 2.6: Data striping [11]

When there are sequential or predictable access patterns to data, this technique can create the virtual effect of having storage devices that can transfer data at a faster rate than any individual disk drive [11].

Communications

As compared to the limited bandwidth available when distributed computing was first emerging, the rapid growth in communication capacity among machines today makes Grid computing practical. Therefore, not surprisingly, another important resource of Grid is data communication capacity. This includes communication within the Grid and external to the Grid.

Software and licenses

The Grid may have software installed that may be too expensive to install on every Grid machine. Using a Grid, the jobs requiring this software are sent to the particular machines on which this software happens to be installed. When the licensing fees are significant, this approach can save significant expenses for an organization [11].

Special equipment, capacities, architectures, policies

Platforms on the Grid will often have different architectures, operating systems, devices, capacities, and equipment. Each of these items represents a different kind of resource that the Grid can use as criteria for assigning jobs to machines.

Jobs and applications

Although various kinds of resources on the Grid may be shared and used, they are usually accessed via an executing "application" or "job [11].

Scheduling, reservation, and scavenging

The Grid system is responsible for sending a job to a given machine to be executed. In the simplest of Grid systems, the user may select a machine suitable for running his job and then execute a Grid command that sends the job to the selected machine.

More advanced Grid systems would include a job "scheduler" of some kind that automatically finds the most appropriate machine on which to run any given job that is waiting to be executed.

The term "scheduling" is not to be confused with "reservation" of resources in advance to improve the quality of service.

In a “scavenging” Grid system, any machine that becomes idle would typically report its idle status to the Grid management node. This management node would assign to this idle machine the next job which is satisfied by the machine’s resources [11].

Intragrid to Intergrid

As presented in figure 2.7, the simplest Grid consists of just a few machines, all of the same hardware architecture and same operating system, connected on a local network. Some people would call this as a “cluster” implementation rather than a “Grid”.

The next progression would be to include heterogeneous machines. Machines participating in the Grid may include ones from multiple departments but within the same organization. Such a Grid is also referred to as an “intragrid”.

The Grid may grow geographically in an organization that has facilities in different cities. Dedicated communications’ connections may be used among these facilities and the Grid.

Over time, as illustrated in Figure 2.8, a Grid may grow to cross organization boundaries, and may be used to collaborate on projects of common interest. This is known as an “intergrid [11].

2.1.3 Cloud Computing

CLOUD computing is a style of computing in which dynamically scalable and virtualized resources are provided as a service over the internet. The concept generally incorporates combination of the following:

- Infrastructure as a Service (IaaS)
- Platform as a Service (PaaS)
- Software as a Service (SaaS)

The topic of cloud computing is gaining more and more attention in the service research community. The main idea is to make applications available on flexible execution environments primarily located in the Internet.

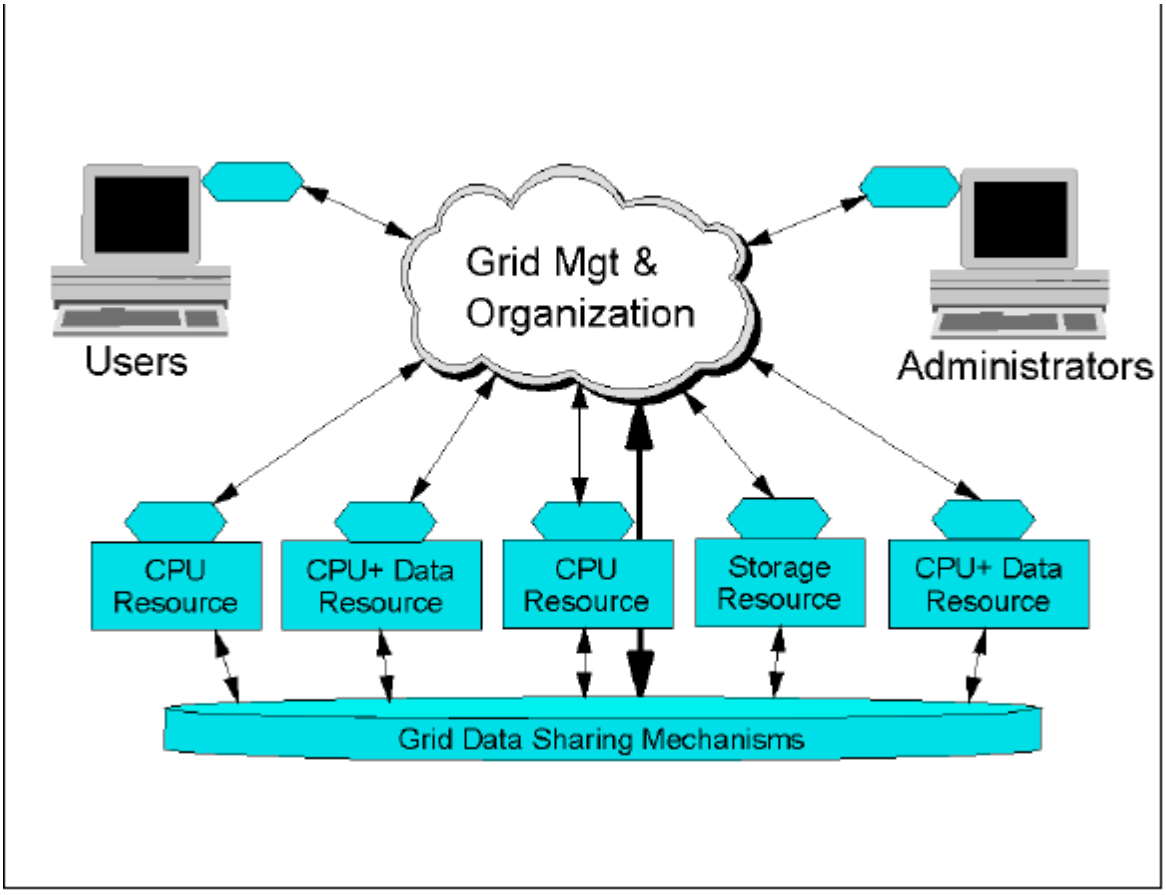

Figure 2.7: A simple Grid [11]

IaaS approach : It refers to the sharing of hardware resources for executing services, typically using virtualization technology. With this approach, potentially multiple users use existing resources. The resources can easily be scaled up when demand increases, and are typically charged for on a per-pay-use basis.

PaaS approach : In this approach, the offering also includes a software execution environment, such as an application server.

PaaS supplies all the resources required to build applications and services completely from the Internet, without having to download or install software.

PaaS services include application design, development, testing, deployment, and hosting.
PaaS generally offers some support to help the creation of user interfaces, and is normally based on HTML or JavaScript [4].

SaaS approach : In this approach, complete applications are hosted on the Internet.

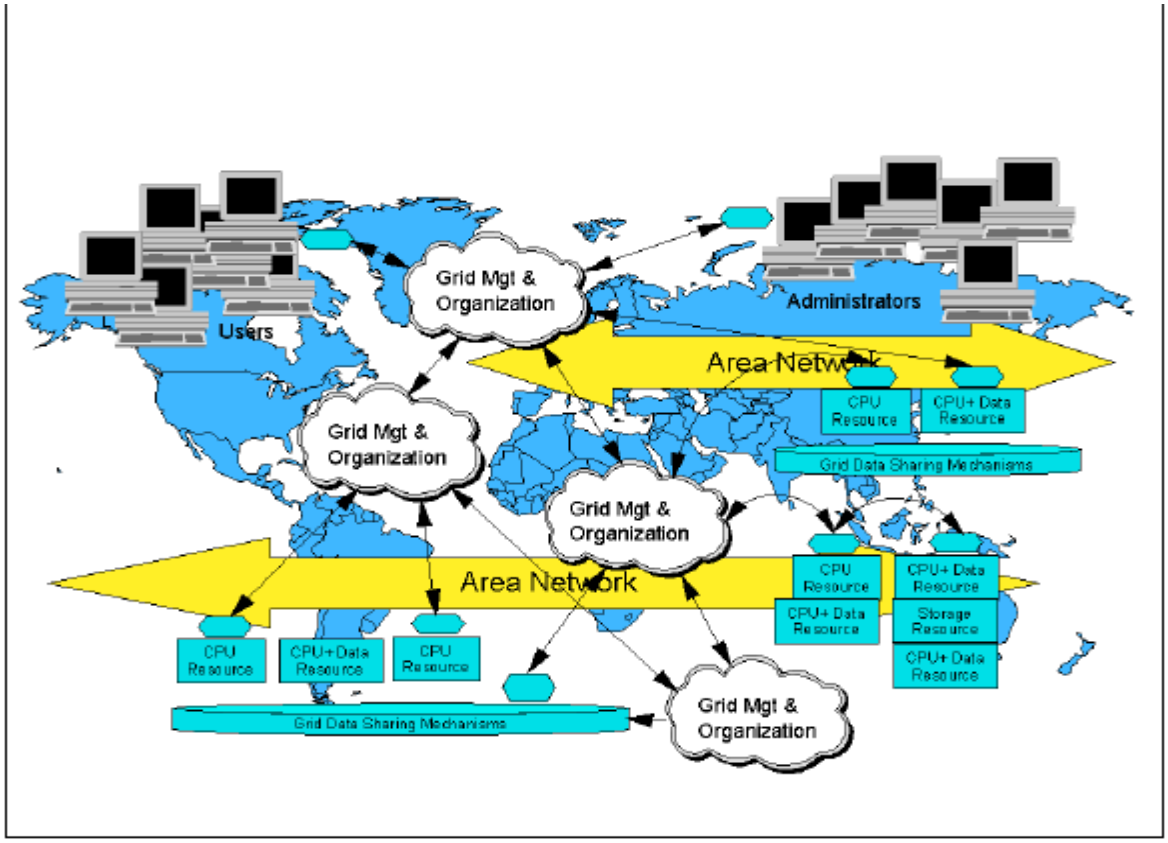

Figure 2.8 : A more complex "Intergrid" [**11]**

2.1.4 Cloud Computing vs Grid Computing

Cloud computing is not a completely new concept, it has an intricate connection to Grid computing paradigm, and other relevant technologies such as cluster computing, utility computing, and distributed systems in general.

2.1.4.1 Clouds, Grids, and distributed systems

Cloud computing not only overlaps with Grid computing, it is indeed evolved out of Grid computing and relies on Grid computing as its backbone and infrastructure support. The evolution has been a result of shift in focus from an infrastructure that delivers storage and

compute resources (such is the case in Grids) to the one that is economy based aiming to deliver more abstract resources and services (such is the case in clouds).

As for utility computing, it is not a new paradigm of computing infrastructure; rather, it is a business model in which computing resources, such as computation and storage, are packaged as metered services. Utility computing is typically implemented using other computing infrastructure (e.g. Grids) with additional accounting and monitoring services.

Figure 2.9 gives an overview of the relationship between Clouds and other domains that it overlaps with. Web 2.0 covers almost the whole spectrum of service-oriented applications, where as Cloud Computing lies at the large-scale side. Supercomputing and Cluster Computing have been primarily focused on traditional non-service applications. Grid Computing overlaps with all these fields where it is generally considered of lesser scale than supercomputers and Clouds.

Cloud Computing is hinting at a future in which we won't compute on local computers, but on centralized facilities operated by third-party compute and storage utilities.

In the mid 1990s, the term Grid was coined to describe technologies that would allow consumers to obtain computing power on demand [12].

2.1.4.2 Comparing Grids and Clouds Side-by-Side In this section, we will compare grids and clouds across a wide variety of perspectives.

2.1.4.2.1 Business Model

In a cloud-based business model, a customer will pay the provider on a consumption basis, very much like the utility companies charge for basic utilities such as electricity, gas, and water.

The business model for Grids (at least that found in academia or government labs) is project-oriented in which the users or community represented by that proposal have certain number of service units (i.e. CPU hours) they can spend [12].

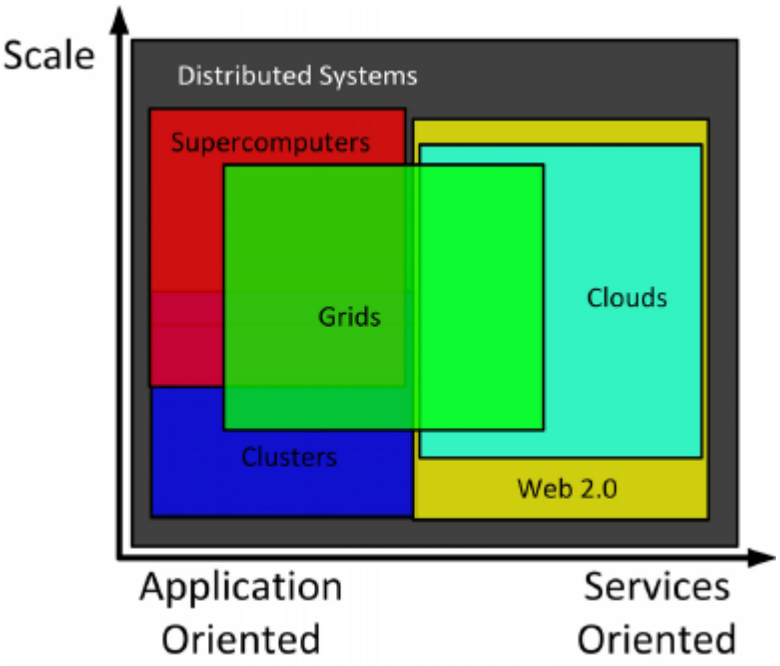

Figure 2.9: Relation of cloud computing with related technologies [12]

2.1.4.2.2 Architecture

Grids provide protocols and services at five different layers as identified in the Grid protocol architecture [12].

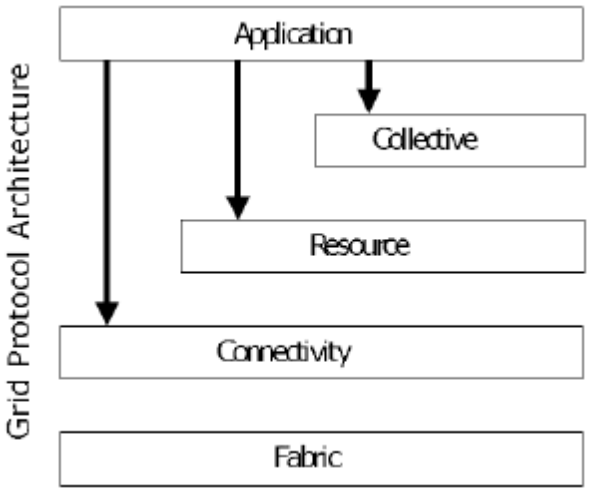

Figure 2.10: Grid Protocol Architecture [12]

At the ***fabric layer***, Grids provide access to different resource types such as compute, storage and network resource, code repository, etc.

The ***connectivity layer*** defines core communication and authentication protocols for easy and secure network transactions.

The ***resource layer*** defines protocols for the publication, discovery, negotiation, monitoring, accounting and payment of sharing operations on individual resources.

The ***collective layer*** captures interactions across collections of resources, directory services such as MDS (Monitoring and Discovery Service).

The ***application layer*** comprises whatever user applications built on top of the above protocols and APIs and operate in VO environments [12].

We define a four-layer architecture for Cloud Computing in comparison to the Grid architecture, as follows:

The *fabric layer* contains the raw hardware level resources, such as compute resources, storage resources, and network resources.

The *unified resource layer* contains resources that have been abstracted/encapsulated (usually by virtualization) so that they can be exposed to upper layer and end users as integrated resources, for instance, a virtual computer/cluster, a logical file system, a database system, etc.

The *platform layer* adds on a collection of specialized tools, middleware and services on top of the unified resources to provide a development and/or deployment platform.

Finally, the *application layer* contains the applications that would run in the Clouds [12].

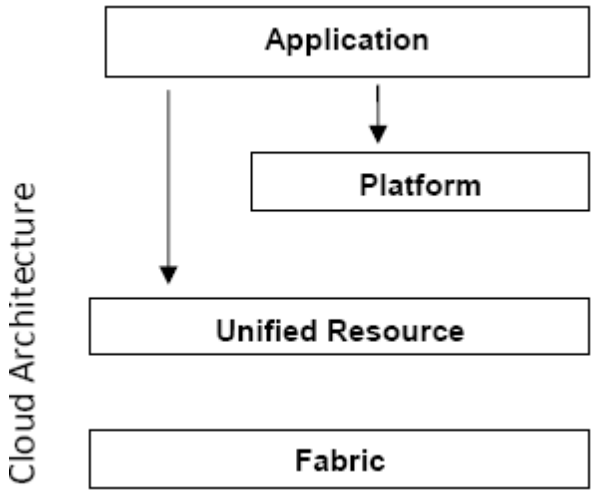

Figure 2.11: Cloud Architecture [12]

2.1.4.2.3 Resource Management

This section describes the resource management found in Grids and Clouds, covering topics such as the compute model, data model, virtualization, monitoring, and provenance.

Compute model : Most Grids use a batch-scheduled compute model, in which a local resource manager (LRM), such as PBS, Condor, SGE manages the compute resources for a Grid site, and users submit batch jobs (via GRAM) to request some resources for some time.

Cloud Computing compute model will likely look very different, with resources in the Cloud being shared by all users at the same time (in contrast to dedicated resources governed by a queuing system) [12].

Data model : We envision that the next-generation Internet Computing will take the triangle model shown in Figure 2.12. Internet Computing will be centralized around Data, Cloud Computing, as well as Client Computing.

Data Locality : In Grids, data storage usually relies on a shared file systems (e.g. NFS, GPFS, PVFS, Luster),where data locality cannot be easily applied. One approach is to improve schedulers to be data-aware, and to be able to leverage data locality information when scheduling computational tasks and this approach has shown to improve job turn-around time significantly.

Combining compute and data management : Combination of the compute and data resource management is critical as it leverages data locality in access patterns to minimize the amount of data movement and improve end-application performance and scalability.

Virtualization: Because of abstraction and encapsulation, virtualization has become an essential ingredient for almost every cloud.

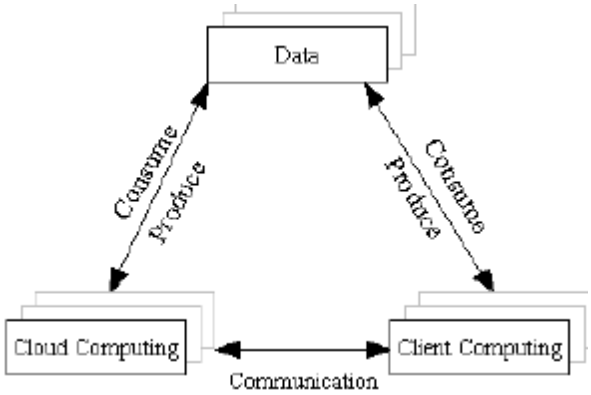

Figure 2.12: The triangle model next-generation Internet Computing [12]

Monitoring : Cloud monitoring is not as straightforward as in Grids, because Grids in general have a different trust model in which users via their identity delegation can access and browse resources at different Grid sites, and Grid resources are not highly abstracted and virtualized as in Clouds.

Provenance : In Grids, provenance management has been in general built into a workflow system, from early pioneers such as Chimera, to modern scientific workflow systems, such as Swift, Kepler and VIEW to support the discovery and reproducibility of scientific results [12].

Provenance is still an unexplored area in Cloud environments, in which we need to deal with even more challenging issues such as tracking data production across different service providers (with different platform visibility and access policies).

2.2 More concepts of Cloud Computing

2.2.1 What Networking of Information can do for Cloud Computing?

Cloud computing is making it possible to separate the process of building an infrastructure for service provisioning from the business of providing end user services. Nowadays, such infrastructures are provided in large data centres and the applications are executed remotely from the users. Networking of Information (NetInf) is an information centric networking paradigm that can support cloud computing by providing new possibilities for network transport and storage [13].

The topic of Cloud Computing is gaining more and more attention in the service research community. The main idea is to make applications available on flexible execution environments primarily located in the Internet.

2.2.2 Cloud Computing Network

"Cloud Computing" is becoming increasingly relevant, as it will enable companies involved in spreading this technology to open the doors to Web3.0.

Cloud computing is thought by some to be an innovative term, recently introduced by the media. The first to give prominence to the term cloud computing (and maybe to coin it) was Google's CEO Eric Schmidt, in late 2006. The term refers to an important and long term trend computing over the Internet [5].

Cloud Vendors

There are number of vendors who offer cloud services. Following are some of the big names in the world of cloud computing with a brief discussion about what they have to offer.

Amazon

Amazon was one of the first companies to offer cloud services to the public. Amazon offers a number of cloud services, including:

- **Elastic Compute Cloud (EC2):** It offers virtual machines and extra CPU cycles for our organization.

- **Simple Storage Service (S3):** It allows us to store items up to 5GB in size in Amazon's virtual storage service.

- **Simple Queue Service (SQS):** It allows our machines to talk to each other using the message-passing API.

- **SimpleDB:** It is a web service for running queries on structured data in real time.

Google

Groups and individuals will likely to get the most out of App Engine by writing a layer of Python that sits between the user and the database.

Microsoft

Microsoft's cloud computing solution is called Windows Azure, an operating system that allows organizations to run Windows applications and store files and data using Microsoft's datacenters [4].

Chapter3

Application Development on Google App Engine

Google App Engine is a cloud environment, and therefore developing an application on Google App Engine helps to explore how the cloud works.

3.1 Gap analysis between the existing work with the targeted work

Cloud is a relatively new technology which evolved after grid computing. Not much work has been done on cloud so far. Therefore, our aim is to demonstrate the working of cloud by developing an application in cloud environment.

3.2 Statement of problem

The aim is to develop an application using python and deploy it on Google App Engine.
Google App Engine lets us run our web applications on Google's infrastructure. App **Engine applications are easy to build, easy to maintain, and easy to scale as our traffic and** data storage needs grow. With App Engine, there are no servers to maintain. We just upload our application, and it's ready to serve our users.

3.3 Justification of problem

The aim to develop and deploy an application on Google App Engine demonstrates the working of new concept “cloud computing”.

3.4 Advantages of deploying applications on Google App Engine

***** Applications will be easy to build and scale.

* There will be no servers to maintain.

* We don’t have to worry about buying servers, load balancers, or DNS tables - Google handles all the heavy listing for us.

* Applications can be built in Python, which is an open source.

* Datastore is maintained by Google, therefore, data security is also not the developer’s headache.

* Cost of development and maintaining applications is greatly reduced.

3.5 Design of application

The application is an opensocial application named “Gifts”.

First of all, the application is created using the Google App Engine Admin console. We access the admin console through our Google account.

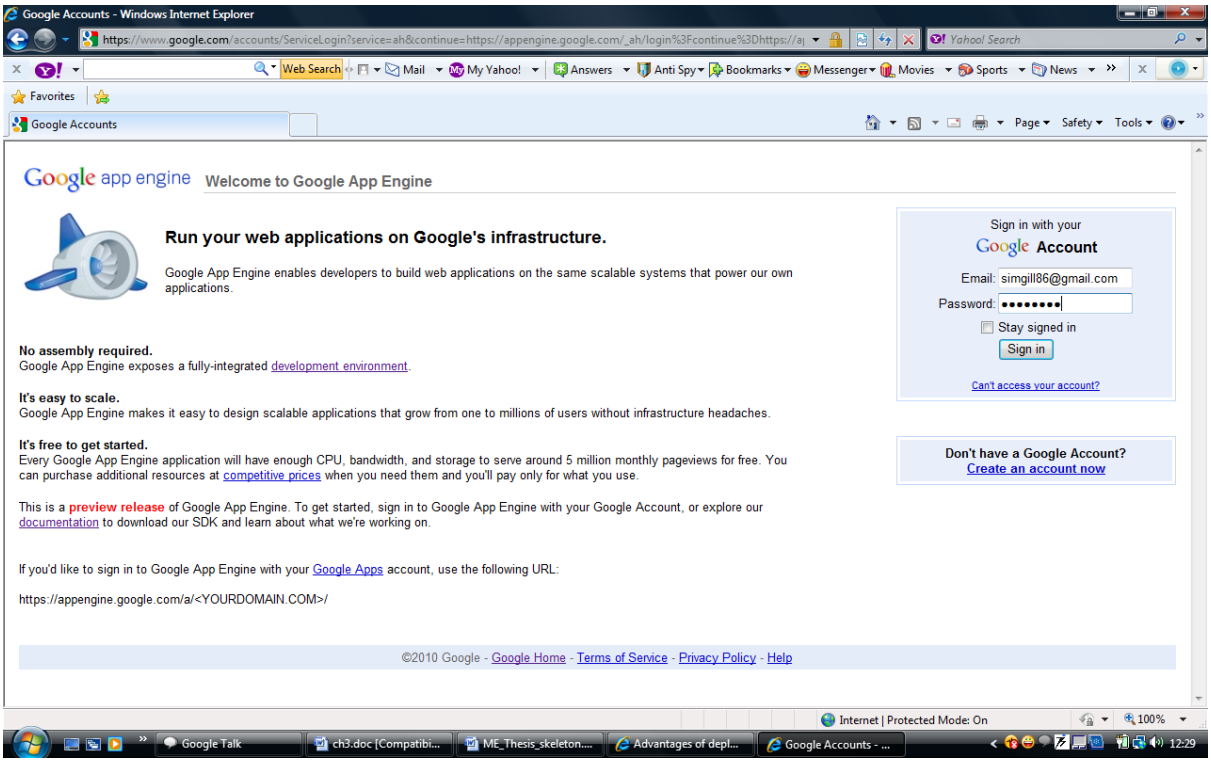

Figure.3.1: Sign in to access admin console

When we click on the “My Applications” link, the list of our applications is displayed. Opensocial-gifts-simthe is our deployed application on Google app engine. Therefore, the url of our application is http://opensocial-gifts-simthe.appspot.com/

Application is built using python (details in chapter 4) and then installed on Orkut sandbox, so that it can be shared with our friends on Orkut.

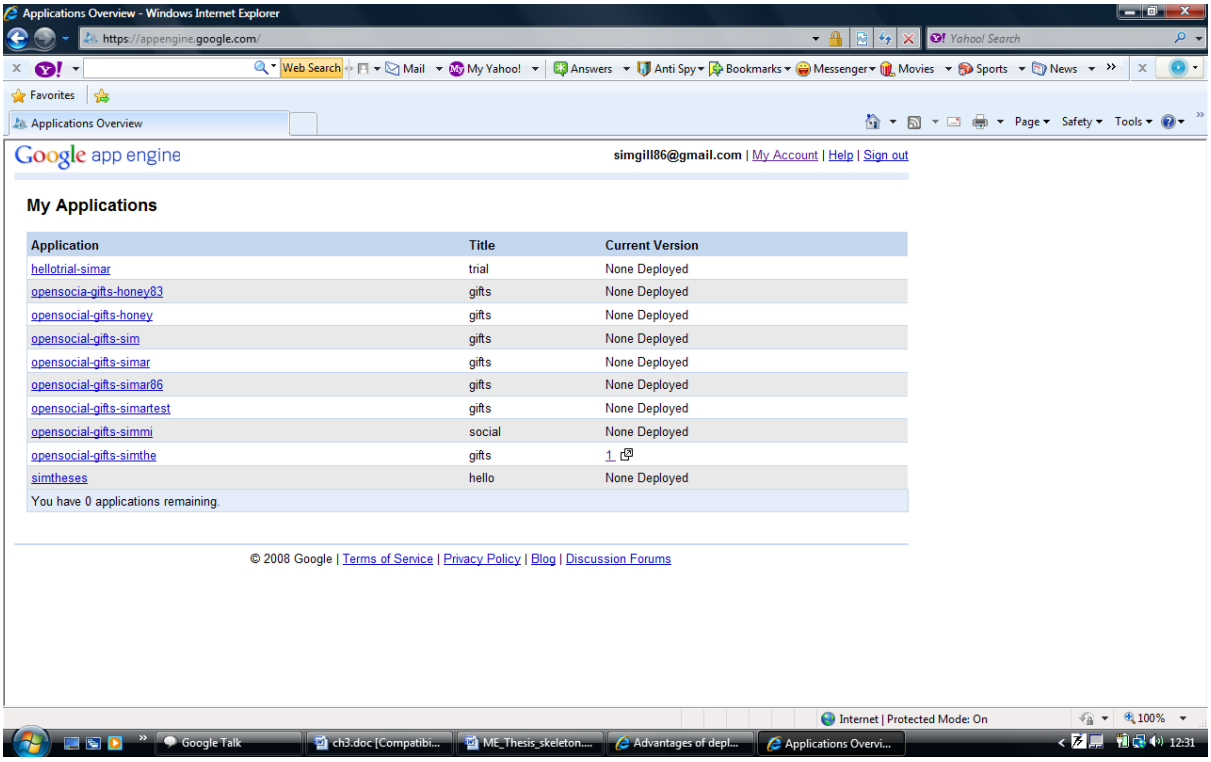

Figure 3.2: List of applications in admin console

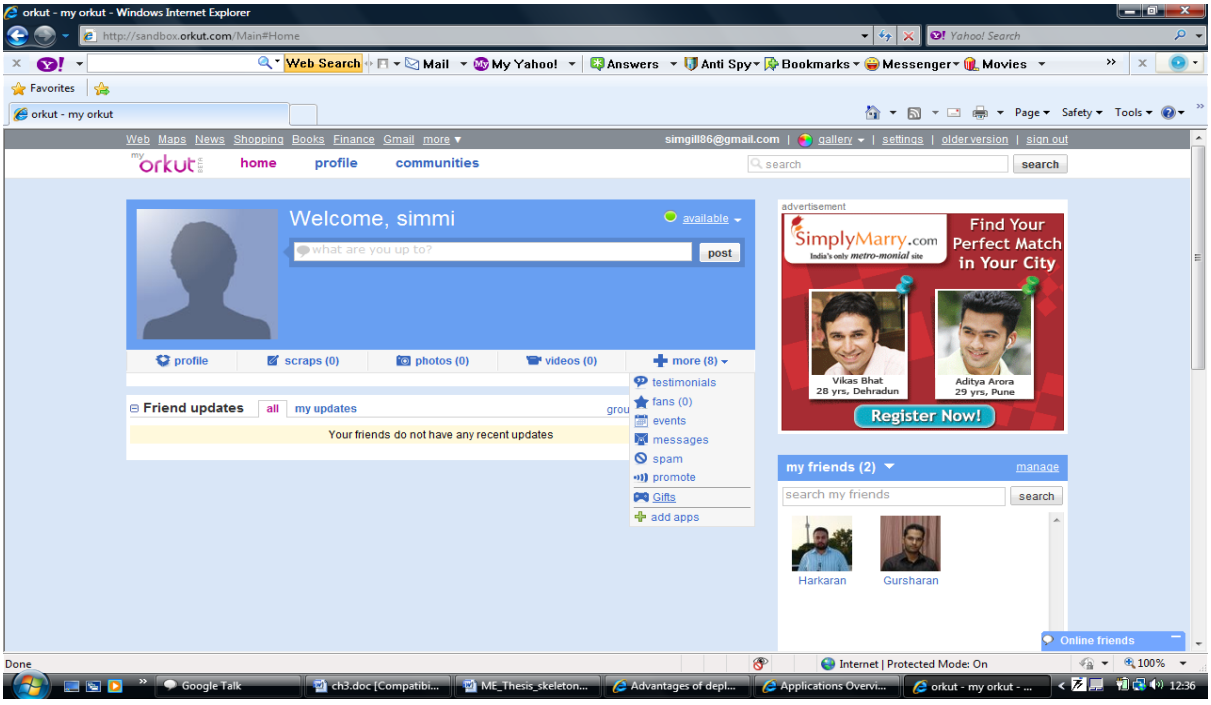

Figure 3.3: Our Orkut home page

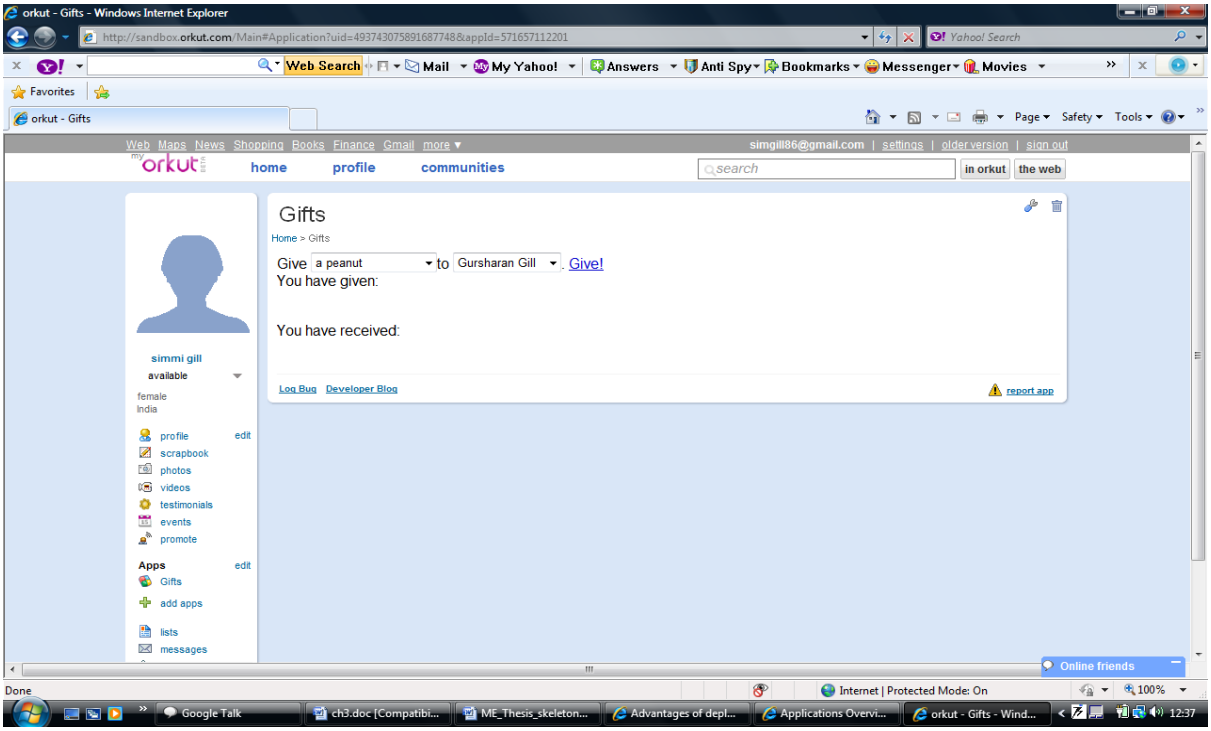

Figure 3.4: Home page of "Gifts" application in Orkut

In this way, gifts can be shared with friends on Orkut.

Google App Engine dashboard

It contains all the details regarding our application, such as the CPU usage, number of requests in last 1 hr, etc..

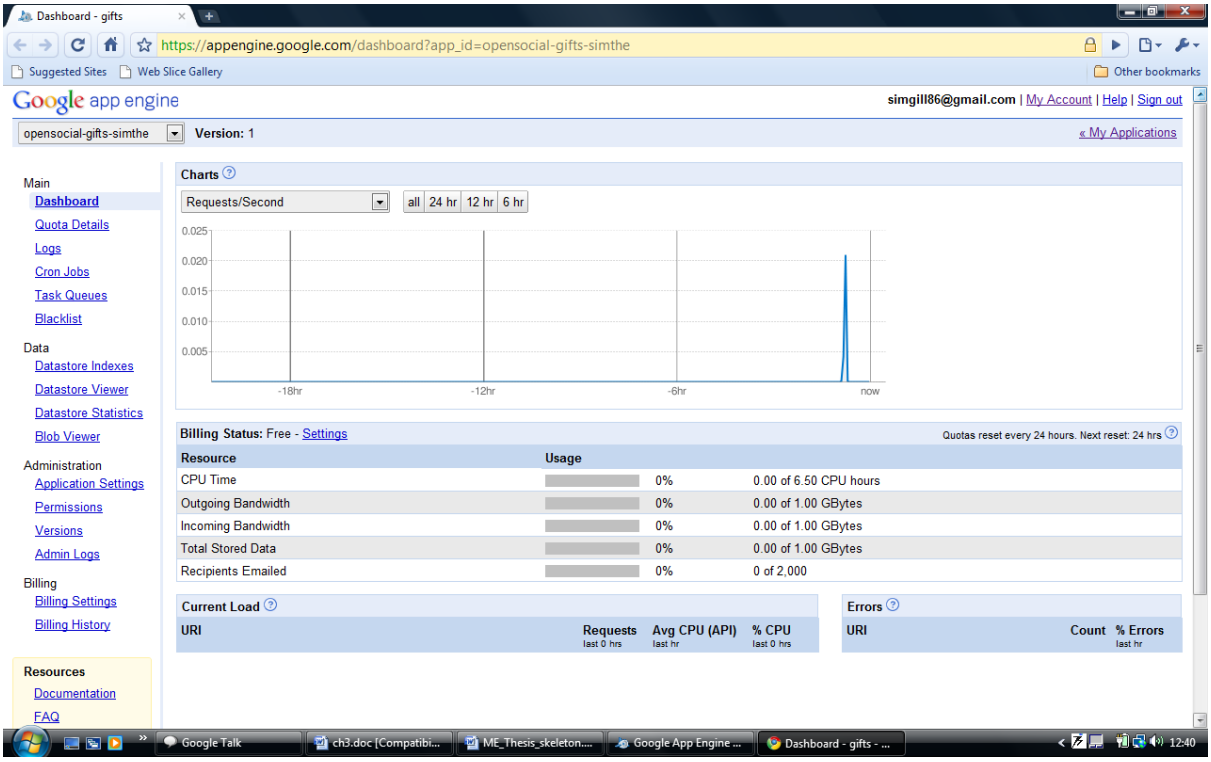

Figure. 3.5: Google App Engine dashboard

Quota details

Each user has a certain free quota on Google App Engine. The user has to pay when this limit exceeds. The quota details for our application are as follows:

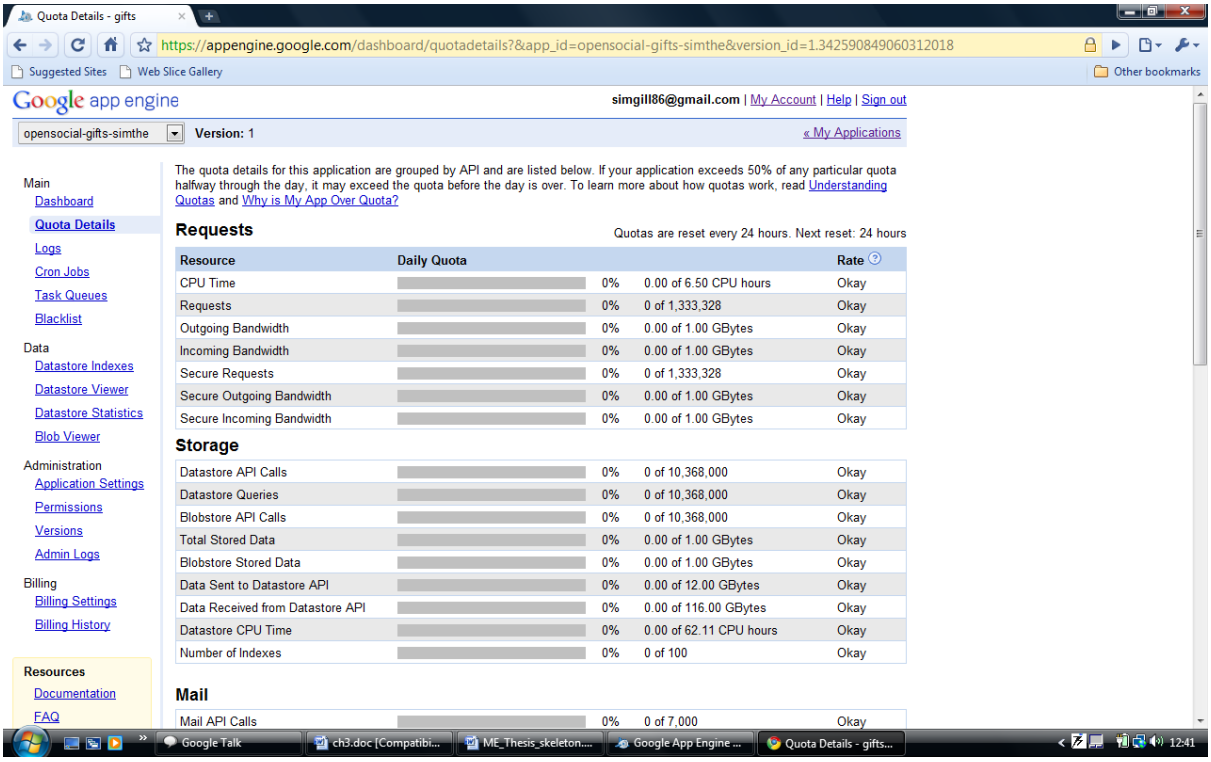

Figure 3.6: Quota details of our application

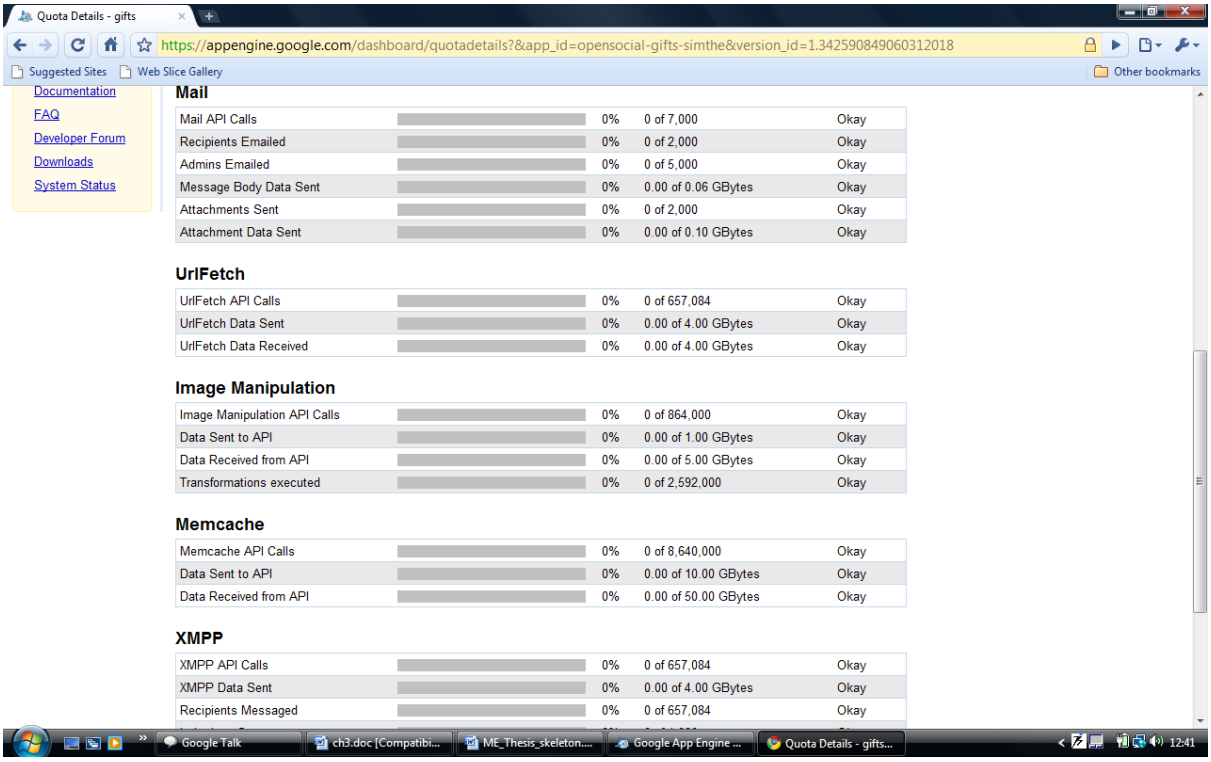

Figure 3.7: Quota details of our application

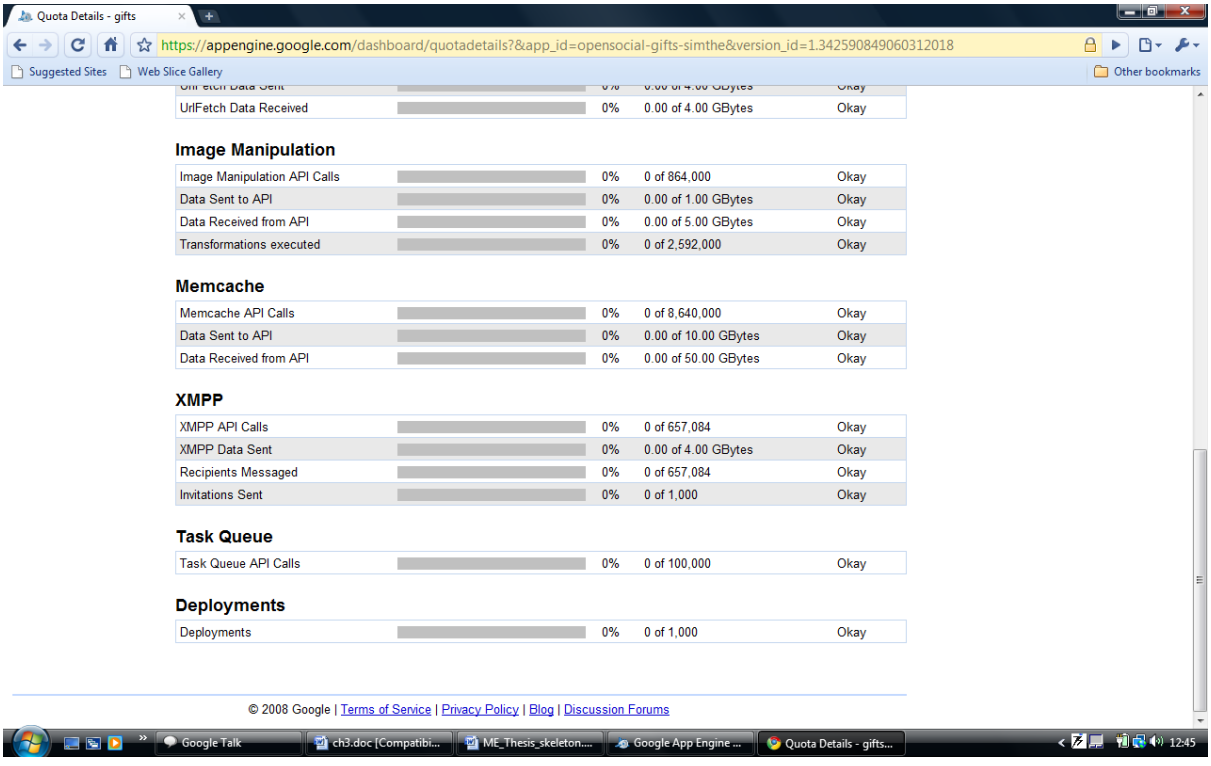

Figure 3.8: Quota details of our application

Google App Engine also maintains the logs of our application.

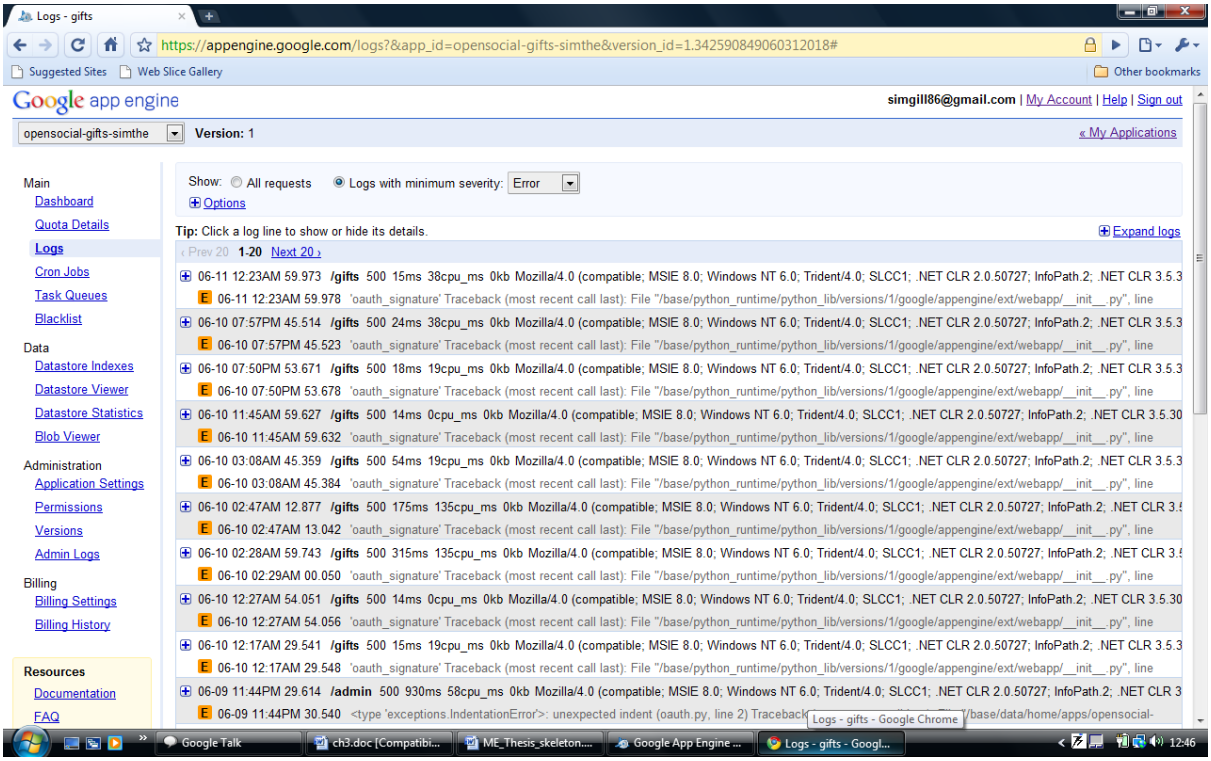

Figure 3.9: Application logs

Datasore Viewer

We can access the datastore of our application through Datastore Viewer in Dashboard in Google App Engine.

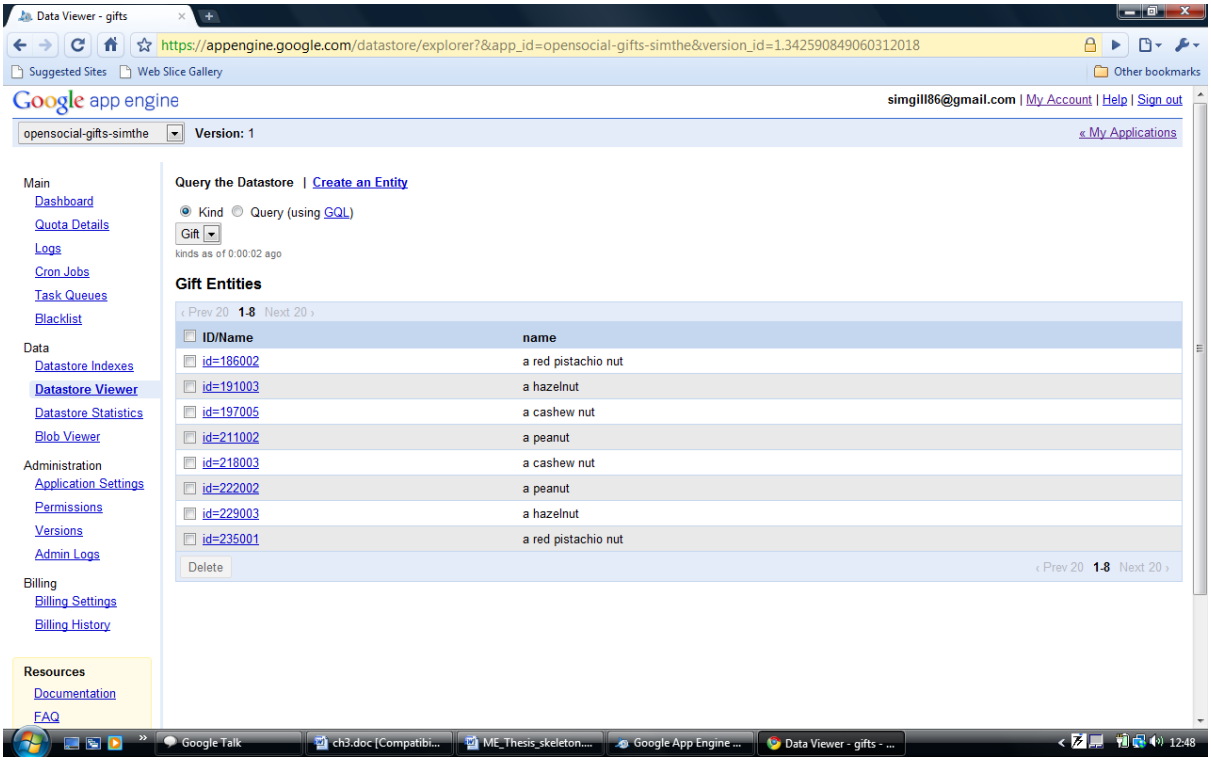

Figure 3.10: Datastore Viewer

Datastore Statistics

We can access Datastore Statistics through dashboard in Google App Engine.

Application Settings

We can access our application settings through dashboard.

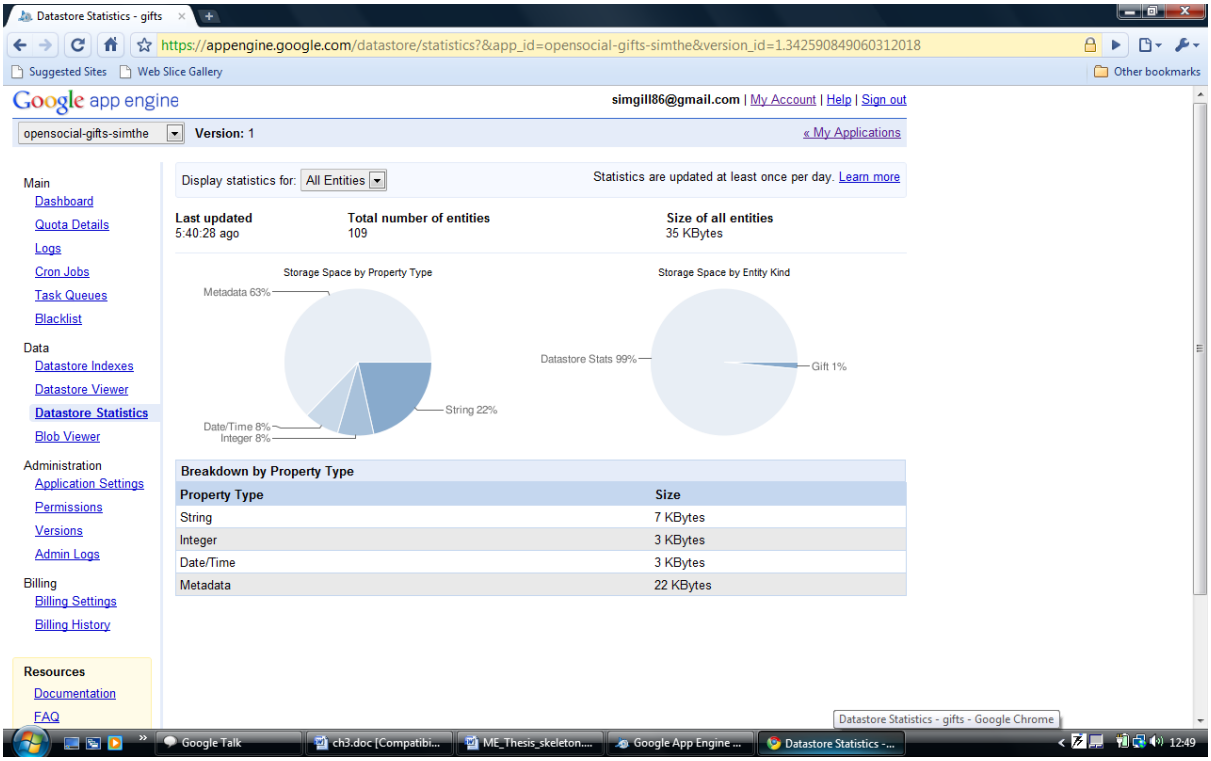

Figure 3.11 :Datastore statistics

Versions

Google App Engine dashboard keeps a record of versions deployed.

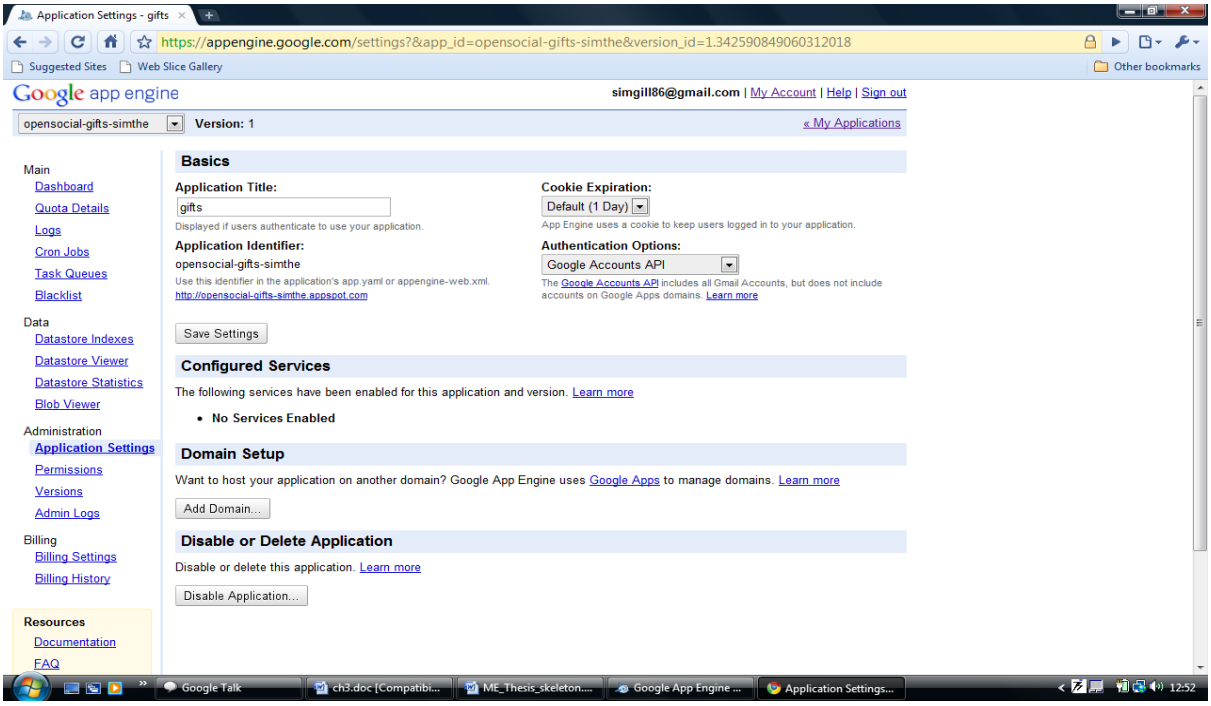

Figure 3.12: Application settings

Permissions

Figure 3.13: Permissions granted.

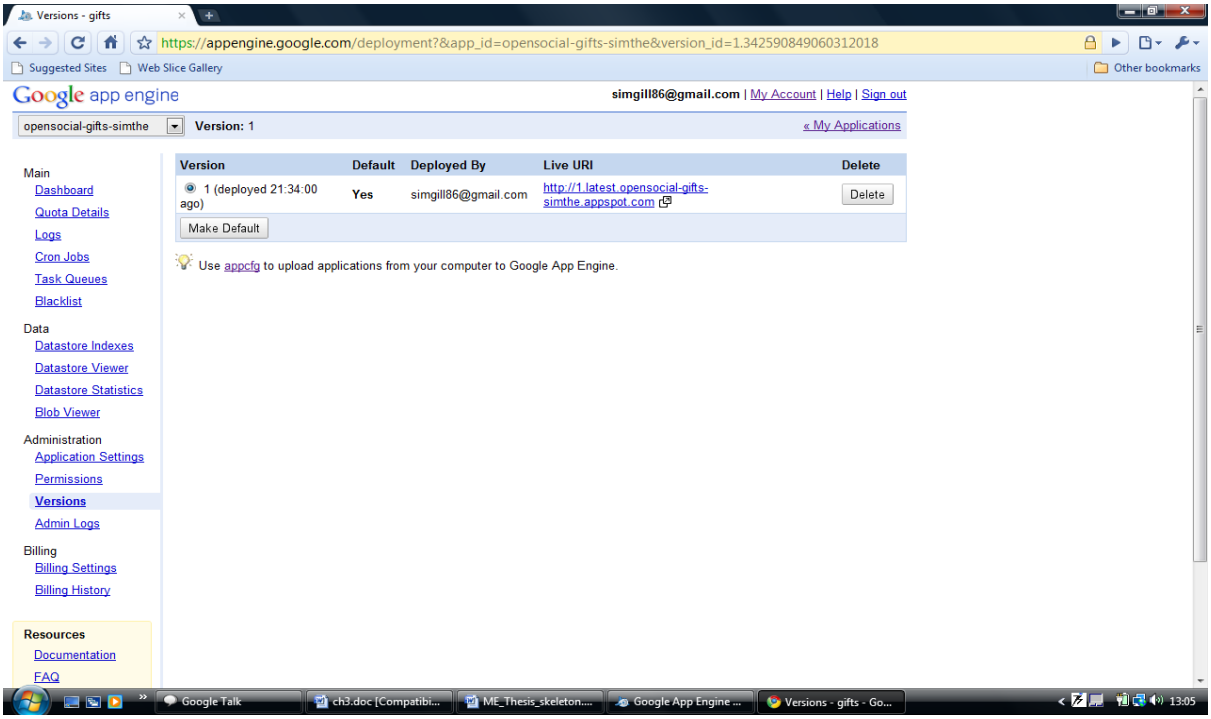

Figure 3.14: Versions of application deployed

Admin Logs

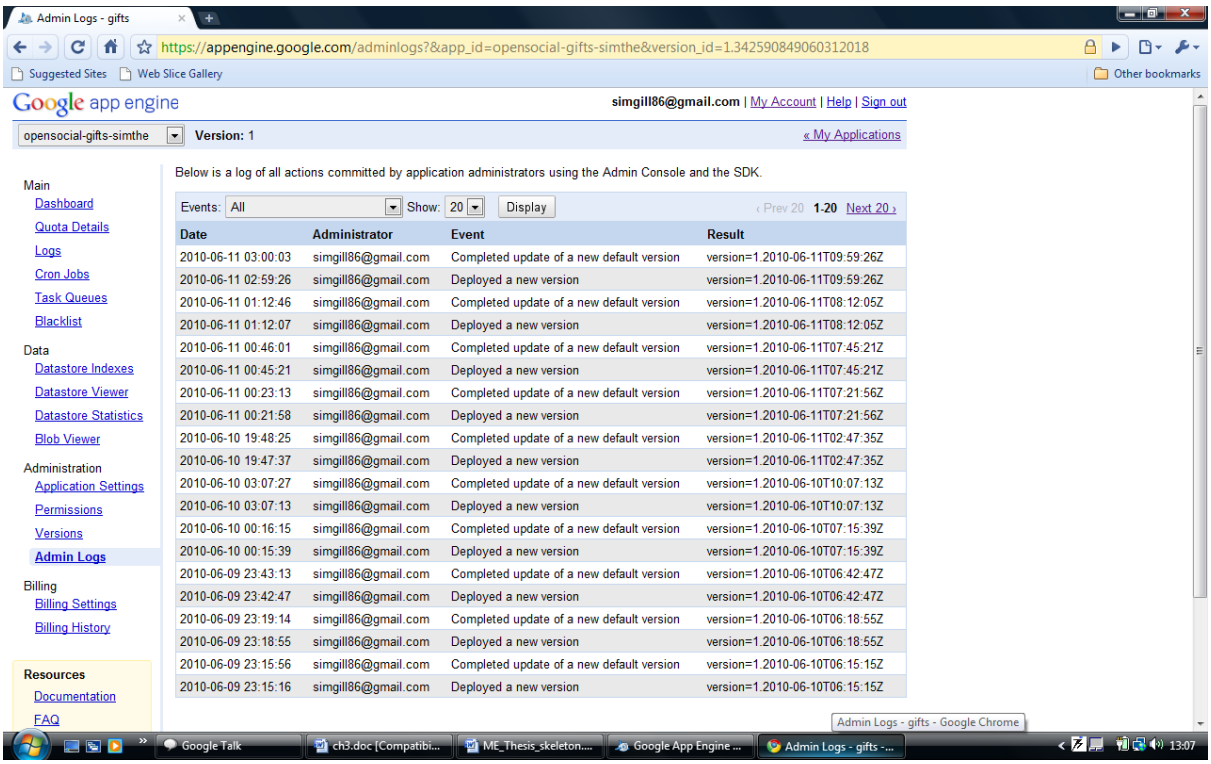

Figure 3.15 :Log of all actions

Billing settings

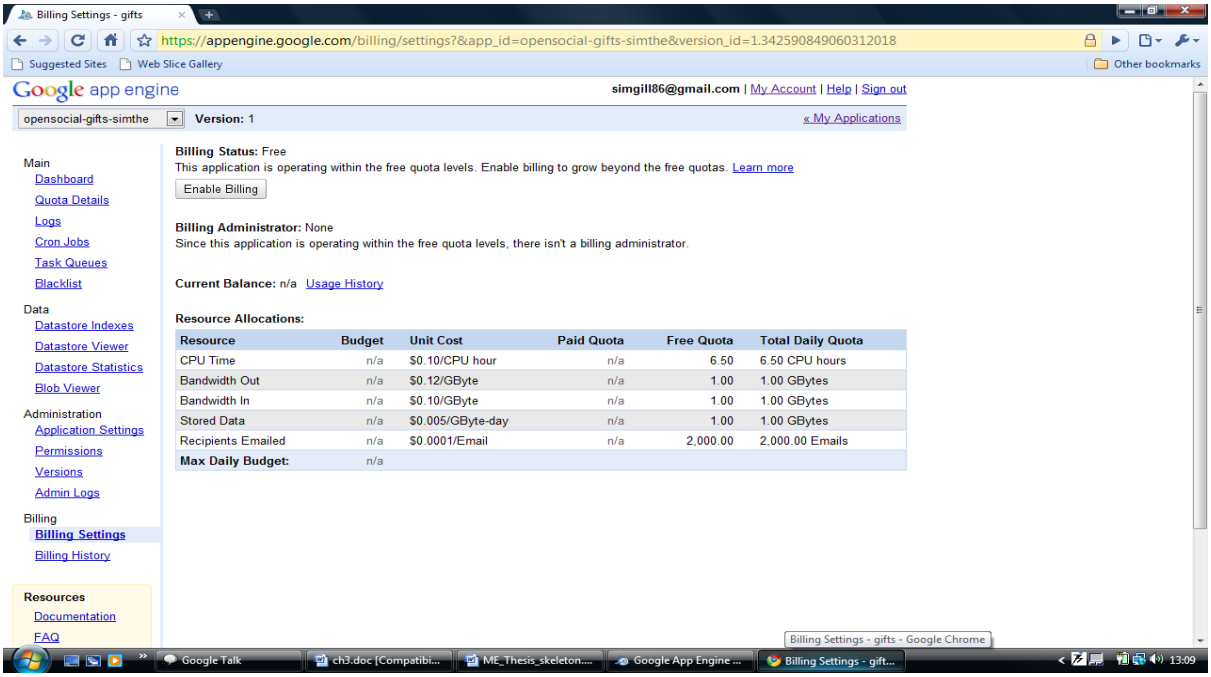

Figure 3.16: Billing details for our application

Billing History

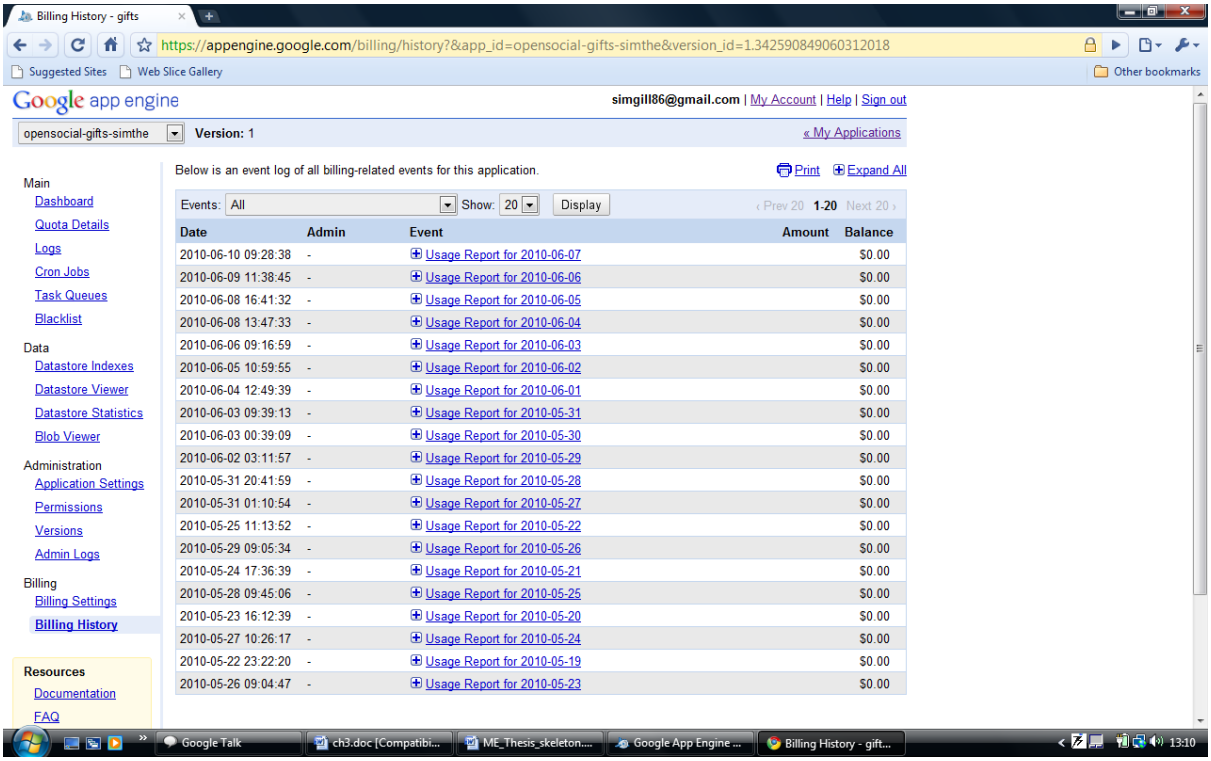

Figure 3.17: Billing history of our application

Chapter 4

Implementation Details

This chapter provides a detailed discussion regarding the implementation of the application.

4.1 Selection of Cloud Environment

Various open source software for cloud computing were explored and finally Google App Engine was used. Following is a brief description of these software:

4.1.1 Nimbus

Nimbus is an open source toolkit that allows us to turn our cluster into an Infrastructure-as-a-Service (IaaS) cloud.

Feature highlights include:

- Two sets of Web Service interfaces: Amazon **EC2 WSDLs** and Grid community **WSRF**
- Implementation based on the **Xen** hypervisor (**KVM** coming soon)
- Can be configured to **use familiar schedulers like PBS or SGE** to schedule virtual machines
- Launches self-configuring **virtual clusters with one click**
- Defines an **extensible architecture** that allows us to customize the software to the needs of our project

Science Clouds

- Science Clouds provide compute cycles in the cloud for scientific communities using Nimbus.
- The Nimbus cloud client allows us to provision customized compute nodes (that we call "workspaces") that we have full control over using a leasing model based on the Amazon's EC2 service.
- The Science Clouds have two objectives:

- to make it easy for scientific and educational projects to experiment with cloud computing
- to enable us to learn how to make cloud computing a useful tool for the scientific community

Available Science clouds

- Nimbus @ University of Chicago
- Stratus @ University of Florida
- Wispy @ Purdue University
- Kupa @ Masaryk University

4.1.2 Eucalyptus

"EUCALYPTUS - Elastic Utility Computing Architecture for Linking Your Programs To Useful Systems - is an open-source software infrastructure for implementing 'cloud computing' on clusters.

The current interface to EUCALYPTUS is compatible with Amazon's EC2 interface, but the infrastructure is designed to support multiple client-side interfaces.

EUCALYPTUS is implemented using commonly-available Linux tools and basic Web-service technologies making it easy to install and maintain."

Some of the key features of Eucalyptus are:

- It installs automatically as part of a Rocks 5 installation
- -- It is modular and extensible, implemented entirely using open-source web service tools
- -- It is interface-compatible with Amazon EC2 and uses the EC2 tools directly
- -- Version 1.0 implements the EC2 features with the exception of static IPs address (planned for a later release)

At the moment, Eucalyptus depends on an open source cluster management software package called Rocks. Rocks is to clusters what Debian, Red Hat, Ubuntu, etc. are to individual Linux machines.
It's a packaging and deployment tool. So to use it (or at least to use it according to our documentation) we need to be using Rocks to manage the software on our clusters.

- Version 1.0 of Eucalyptus, downloadable now, is a feature-limited binary-only beta at this point.
- Eucalyptus was publicly demonstrated at the Open Source Grid and Cluster conference on May 14th.

4.1.3 Hadoop

The Apache Hadoop project develops open-source software for reliable, scalable, distributed computing.

Hadoop Distributed File System (HDFS) is the primary storage system used by Hadoop applications. HDFS creates multiple replicas of data blocks and distributes them on compute nodes throughout a cluster to enable reliable, extremely rapid computations.

The first step in implementation was to install Hadoop on the system.

Installation of hadoop on Linux

Fedorra was installed on the system. The next step was to remove existing Java and install java 1.6, but the commands for doing so didn't work because of some compatibility issues, therefore there was a redirection to install Hadoop on windows.

Installation of Hadoop on Windows

The aim was to make a simplified version of Hadoop cluster that would run locally on the developer's machine.

The first step was to install java 1.6 and Eclipse Europa 3..3.2 on the system as these are the pre-requisites for the Hadoop installation.

4.1.3.1 Installation of java 1.6

Java was downloaded from http://java.sun.com/ and installed on the system. The screenshots for java installation are as follows:

We need to login as a user to download java using sun download manager.

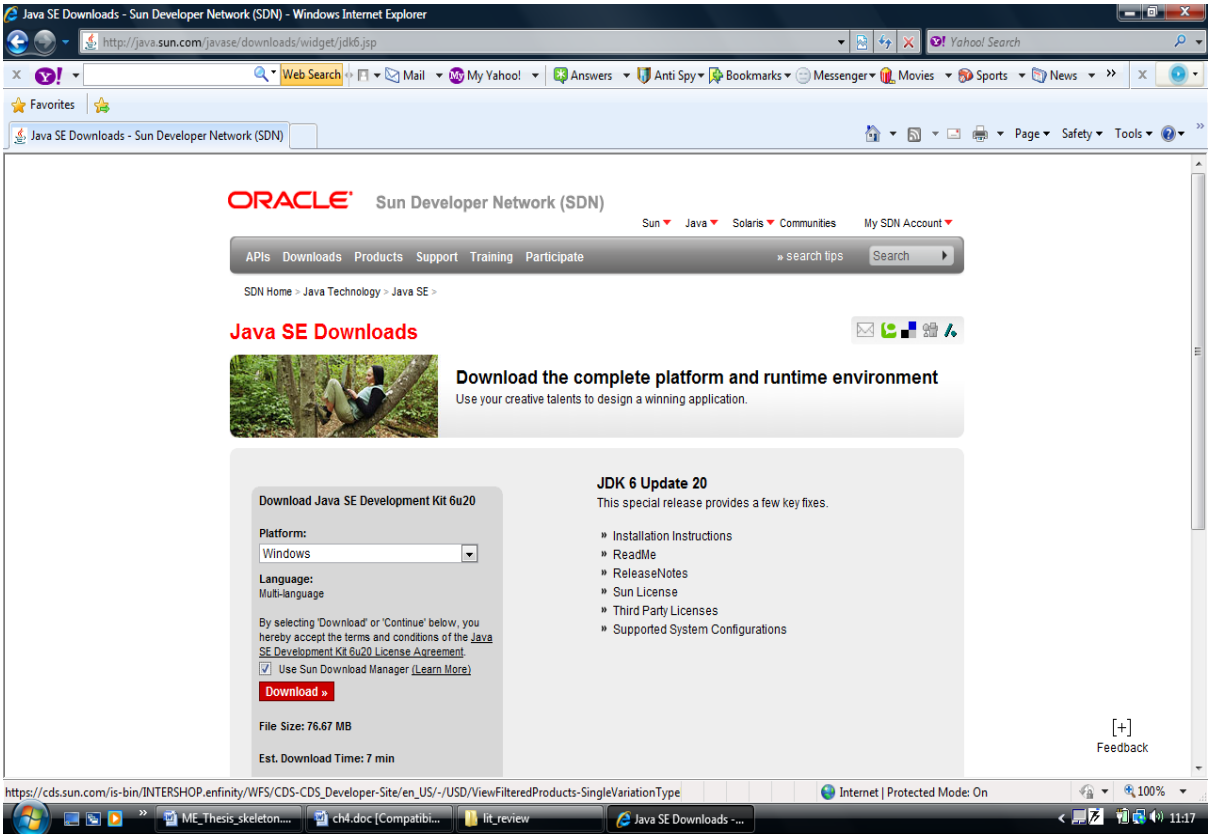

Figure 4.1: The page for downloading java

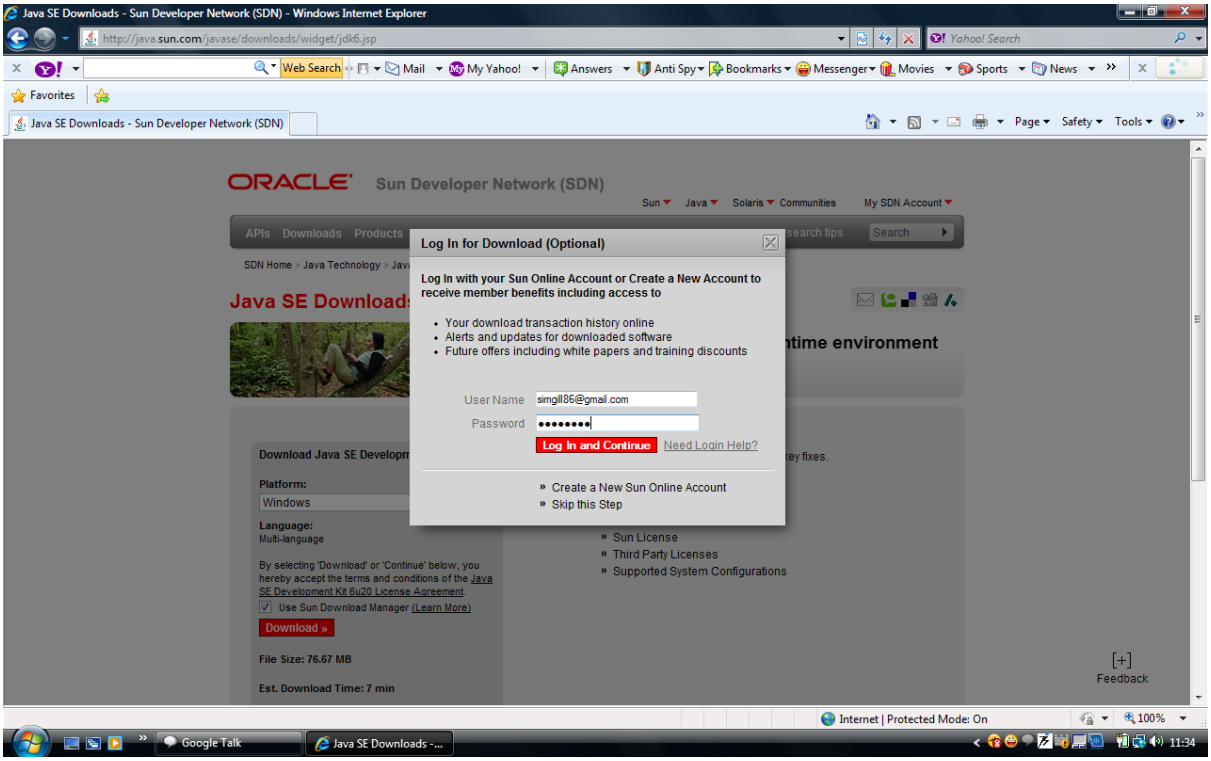

Figure 4.2: Log in page for downloading java

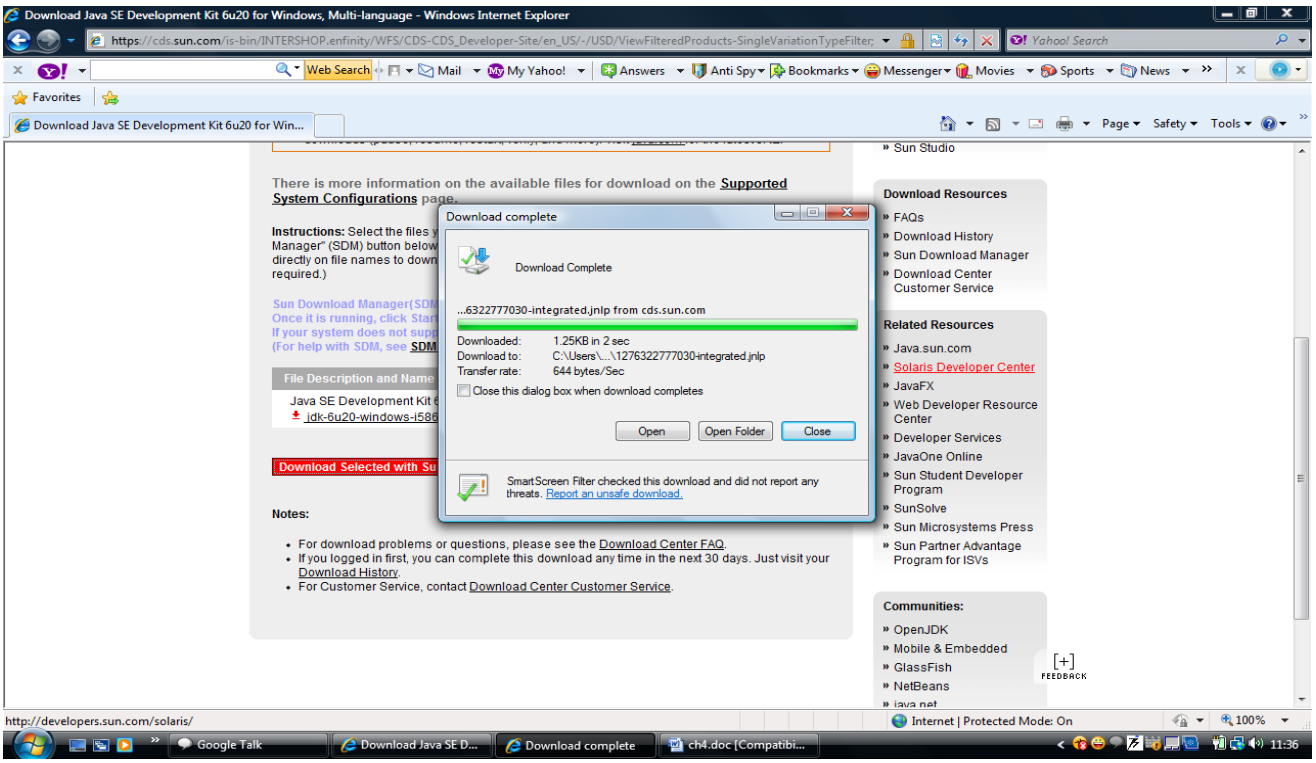

Figure 4.3 Download of java in progress

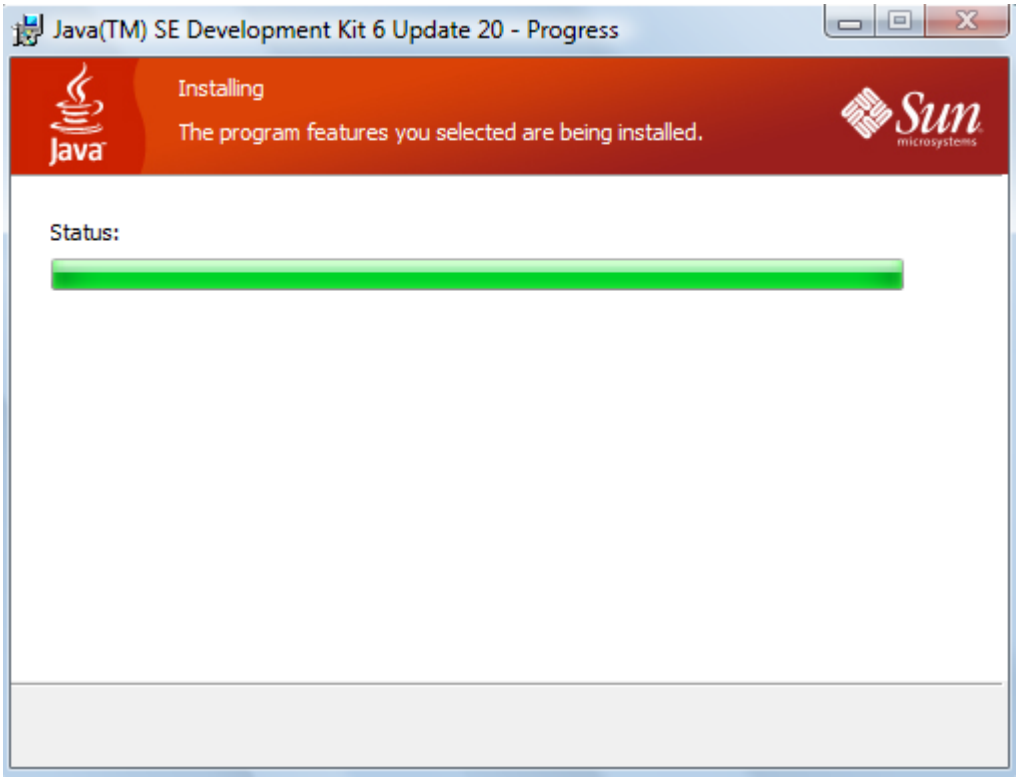

Figure 4.4 :Installation of java in progress

4.1.3.2 Installing Eclipse Europa 3.3.2

Eclipse was downloaded from www.eclipse.org .The screenshots are as follows:

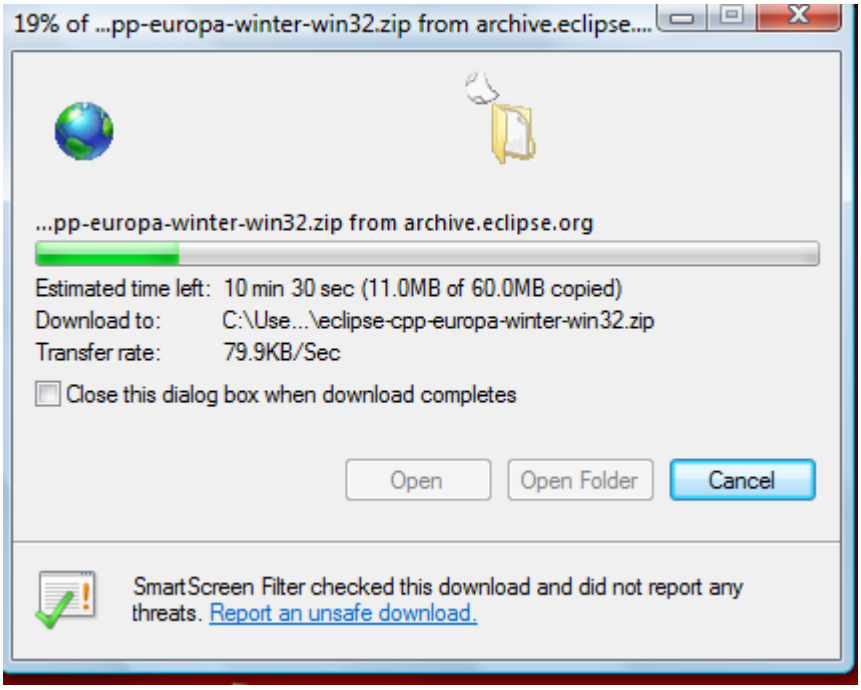

Figure 4.5: Download in progress

We need to uncompress the downloaded eclipse-cpp-europa-winter-win32.zip and then run the eclipse.exe file.

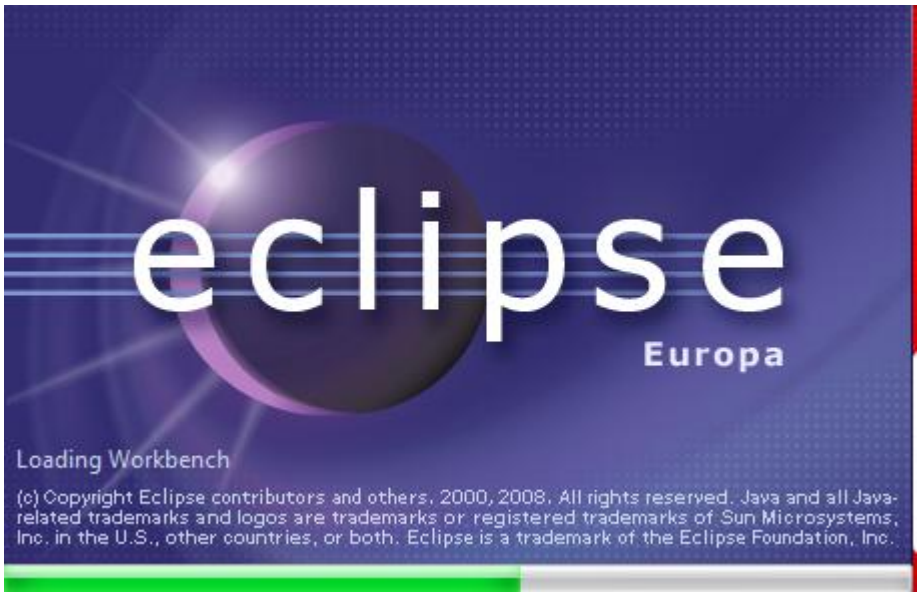

Figure4.6: Workspace launcher for eclipse

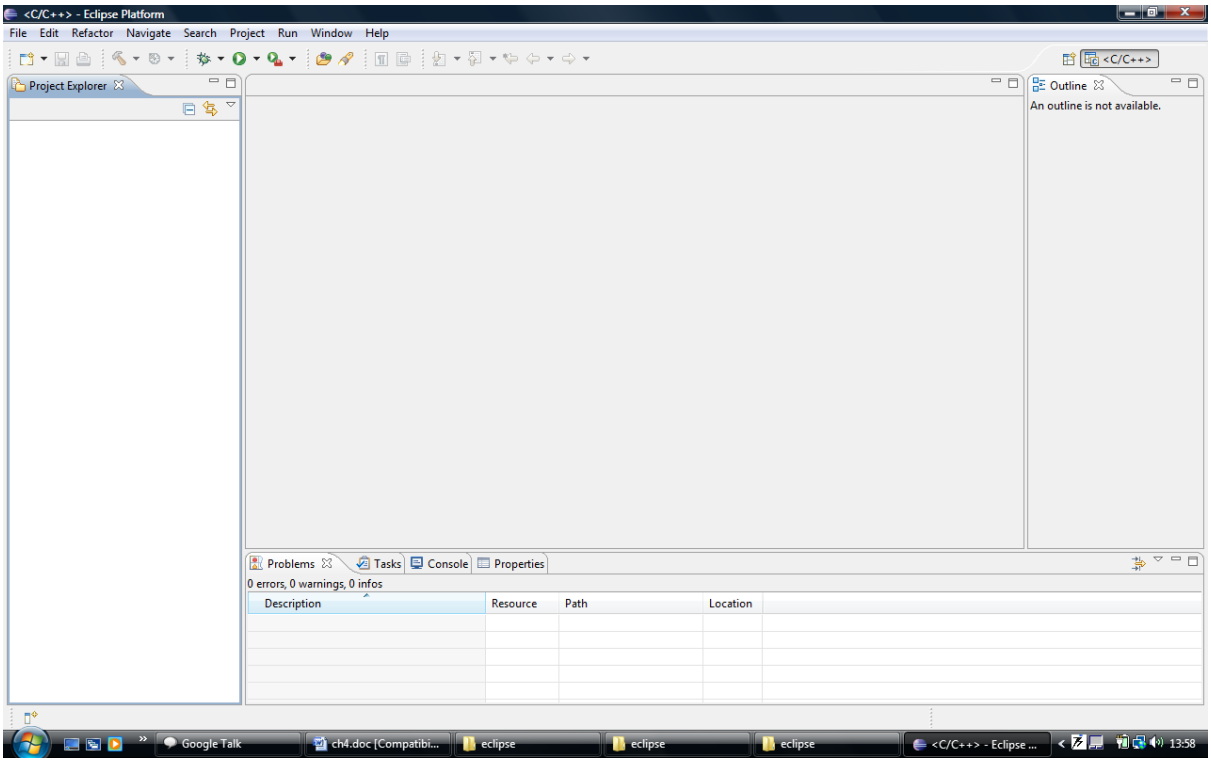

Figure 4.7: Eclipse worksheet

Eclipse worksheet is opened after running the launcher.

4.1.3.3 Installing Cygwin

After installing the prerequisite software, the next step is to install the Cygwin environment. Cygwin is a set of Unix packages ported to Microsoft Windows. It is needed to run the scripts supplied with Hadoop because they are all written for the Unix platform.

To install the cygwin environment we have to follow these steps:

1. Download cygwin installer from http://www.cygwin.com.
2. Run the downloaded file.

After Cygwin installation, we have to configure the environment variables and then configure ssh deamon. There was som problem in configuration due to some compatibility issues, therefore there was a shift to Google App Engine.

So finally, we developed our application on Google app Engine, for which we installed Python 2.5 as a prerequisite.

Python was downloaded from www.python.org/ and installed on the system.

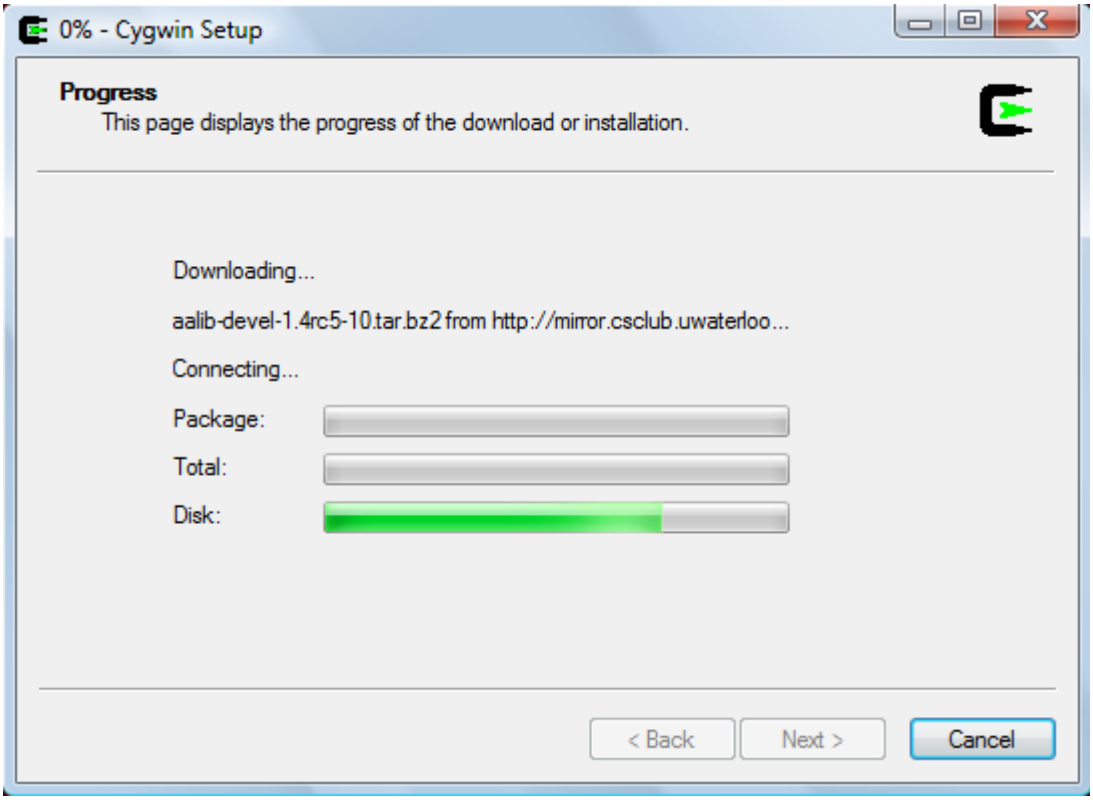

Figure 4.8: Cygwin Setup in progress

4.2 Installation of Google App Engine

Google App Engine was downloaded from http://code.google.com and installed on the system.

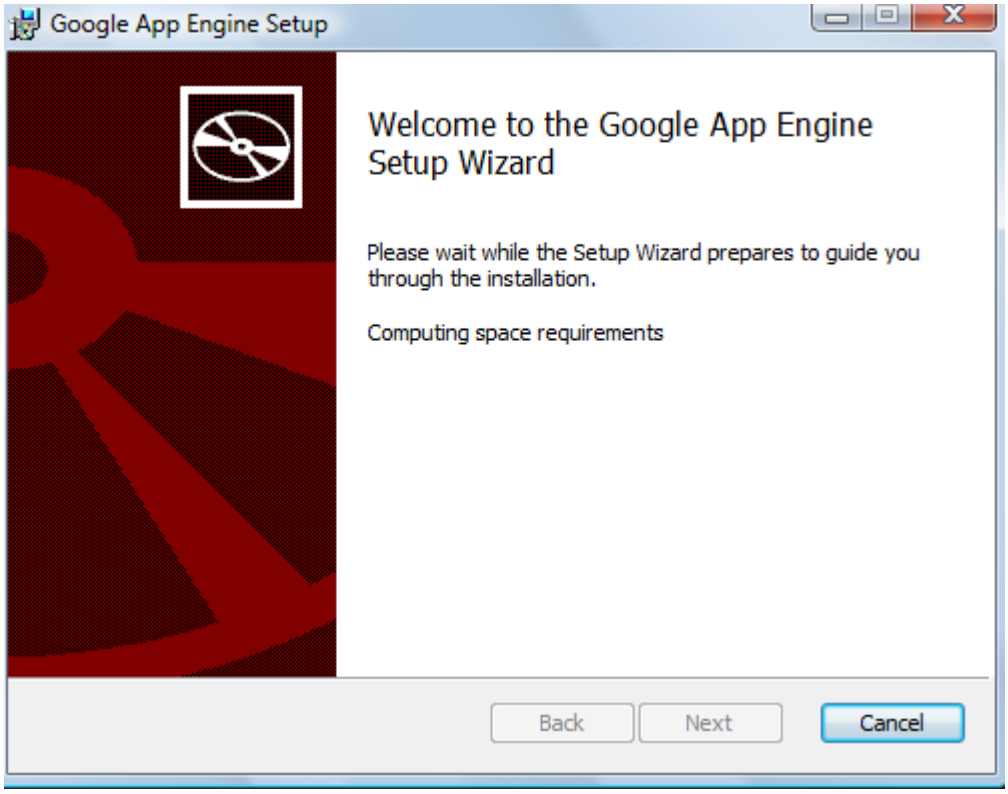

Figure 4.9: Google App Engine Setup

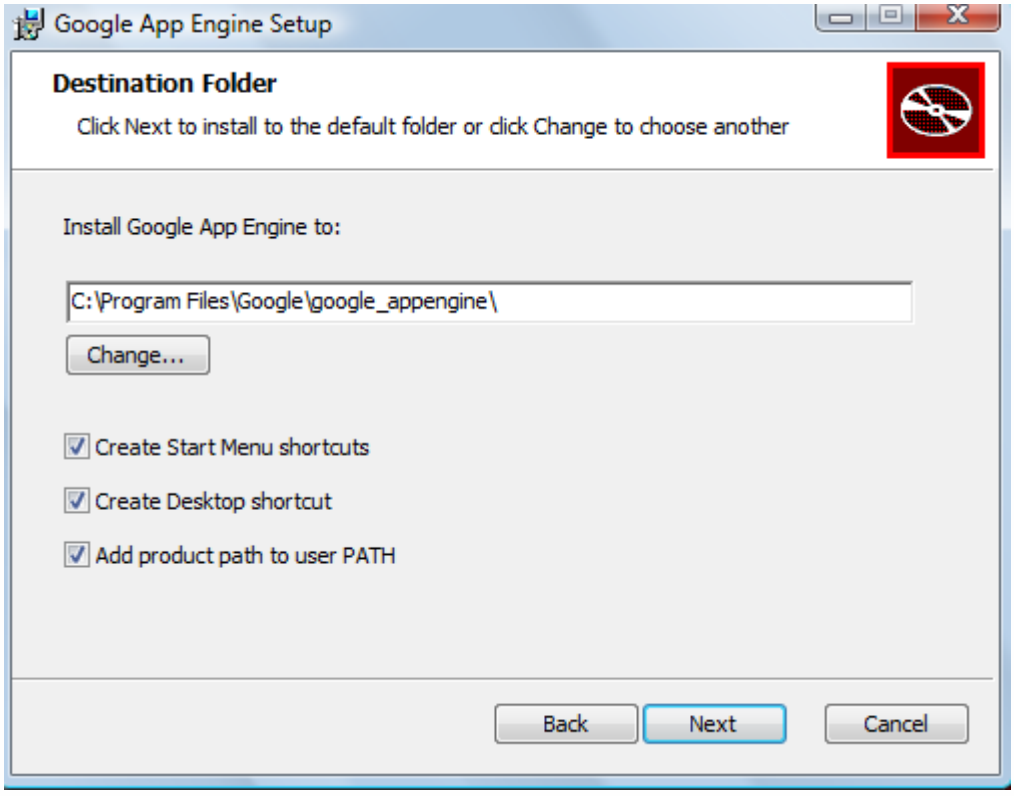

Figure 4.10: Selection of destination folder

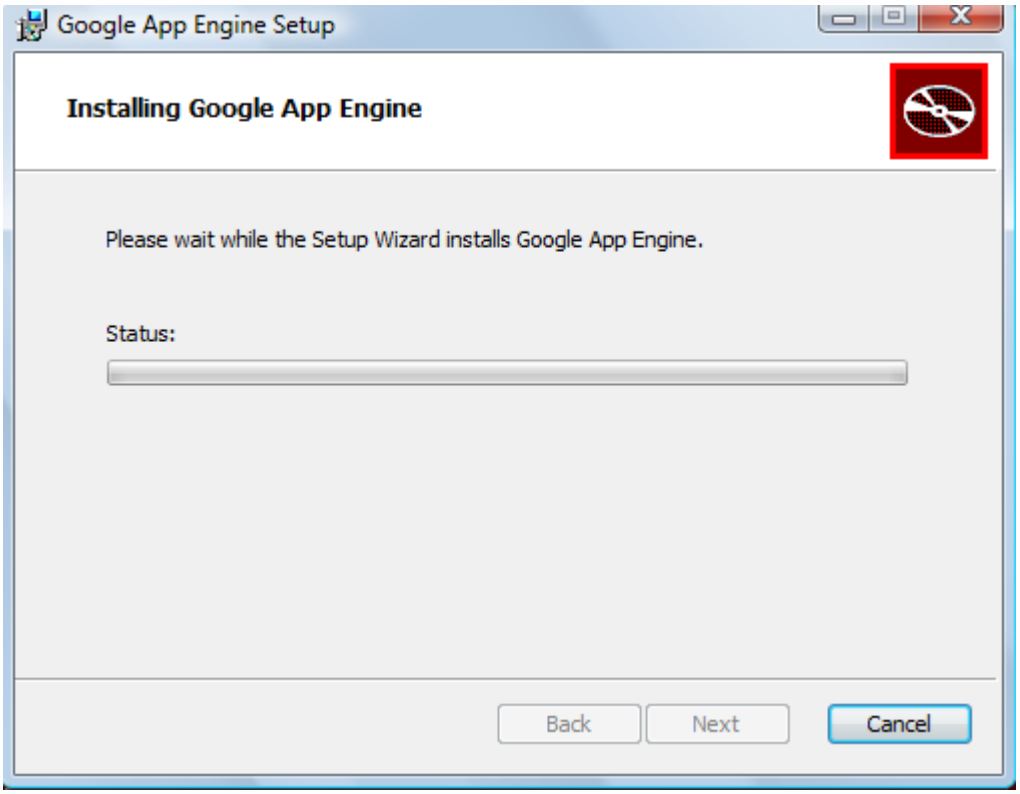

Figure 4.11: Installation in progress

4.3 Development of Application

Google App Engine home page is as in Figure 4.12.

We create our application directory by clicking on Create new application in File menu.

After this, application is registered on Google App Engine admin console. The name of our application directory is opensocial-gifts-simthe, therefore, the url of our application becomes http://opensocial-gifts-simthe.appspot.com.

Coding is done in python language. We can run the application by clicking the "Run" on Google App Engine.

We can then browse the application from browser using the link http://opensocial-gifts-simthe.appspot.com.

We use Orkut website as the container for application. The application is installed on the Orkut sandbox.

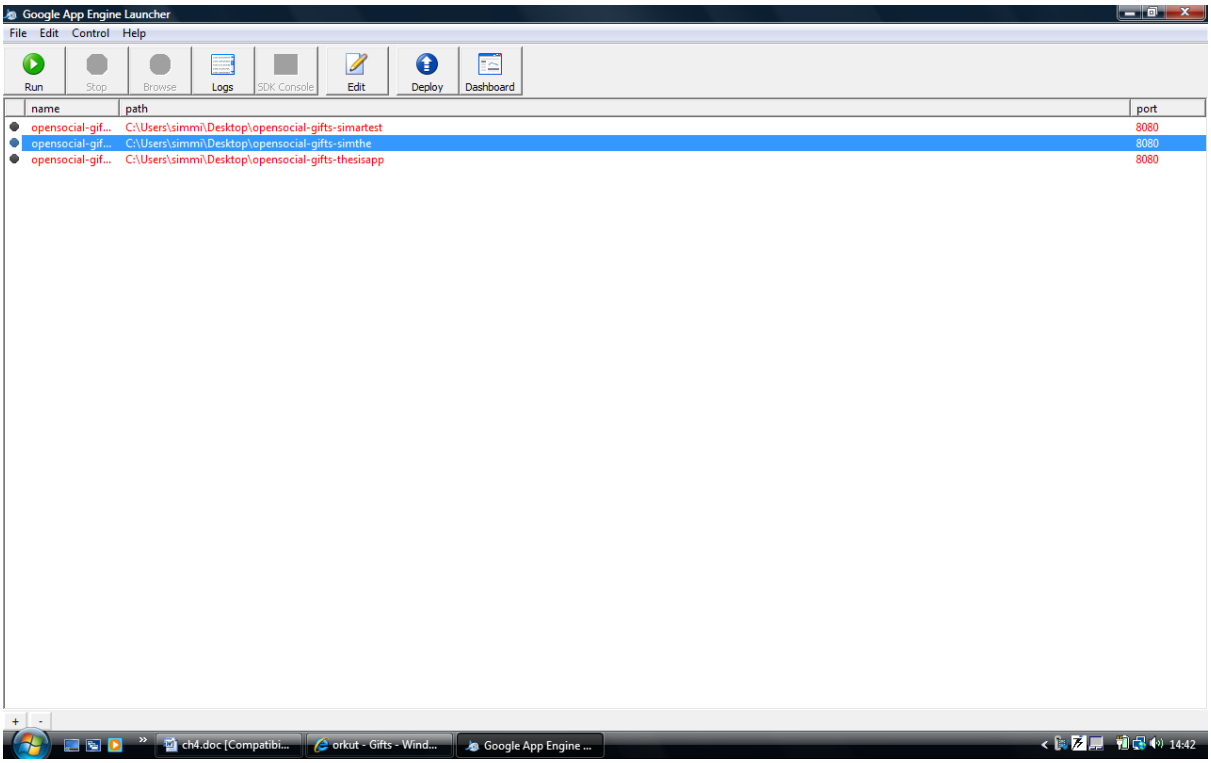

Figure 4.12 Google App Engine

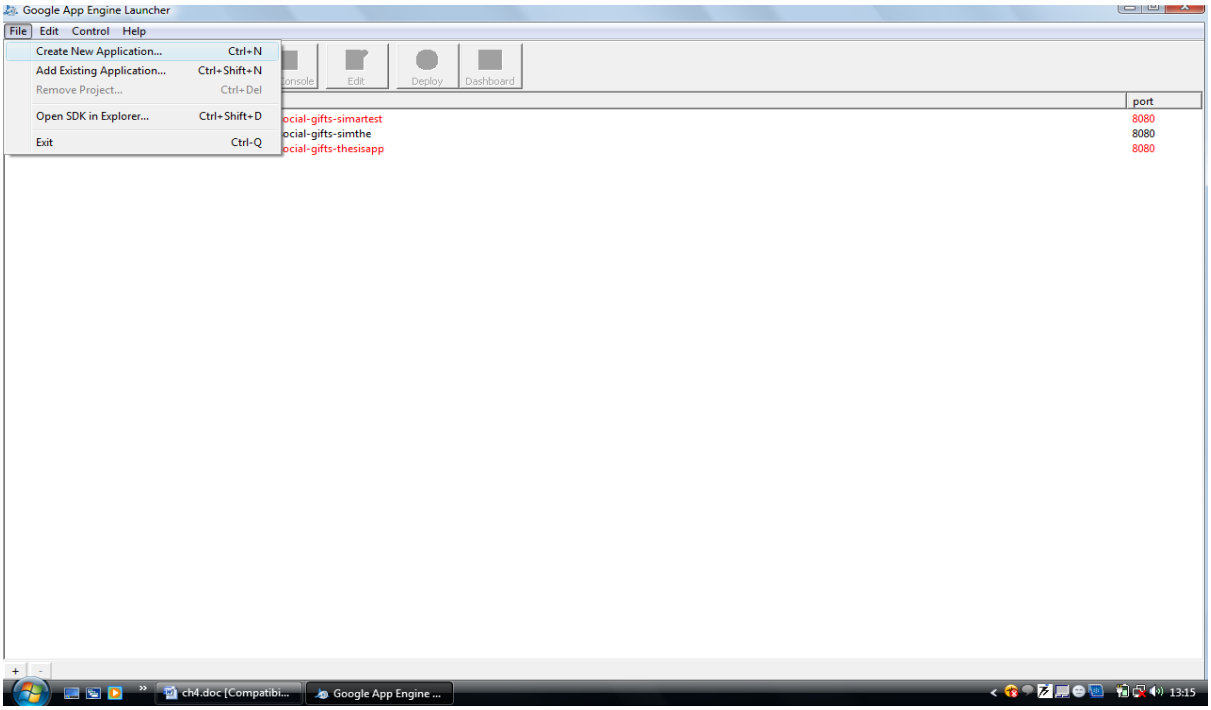

Figure 4.13: Creation of new application

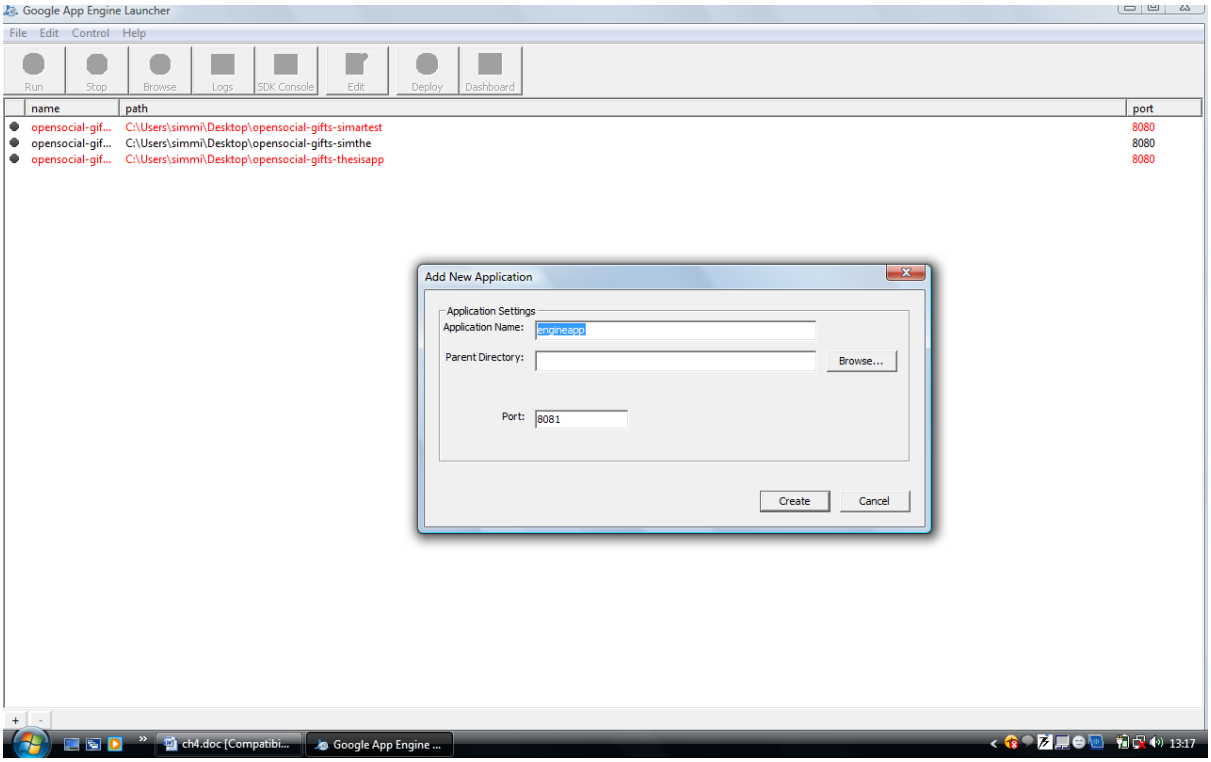

Fig. 4.14: Selection of application settings

Chapter 5

Testing

After running our application from Google App Engine, we test our application by visiting the application url.

5.1 Browsing the Admin page

We get the following screen when we browse the url http://opensocial-gifts-simthe.appspot.com/admin/ :

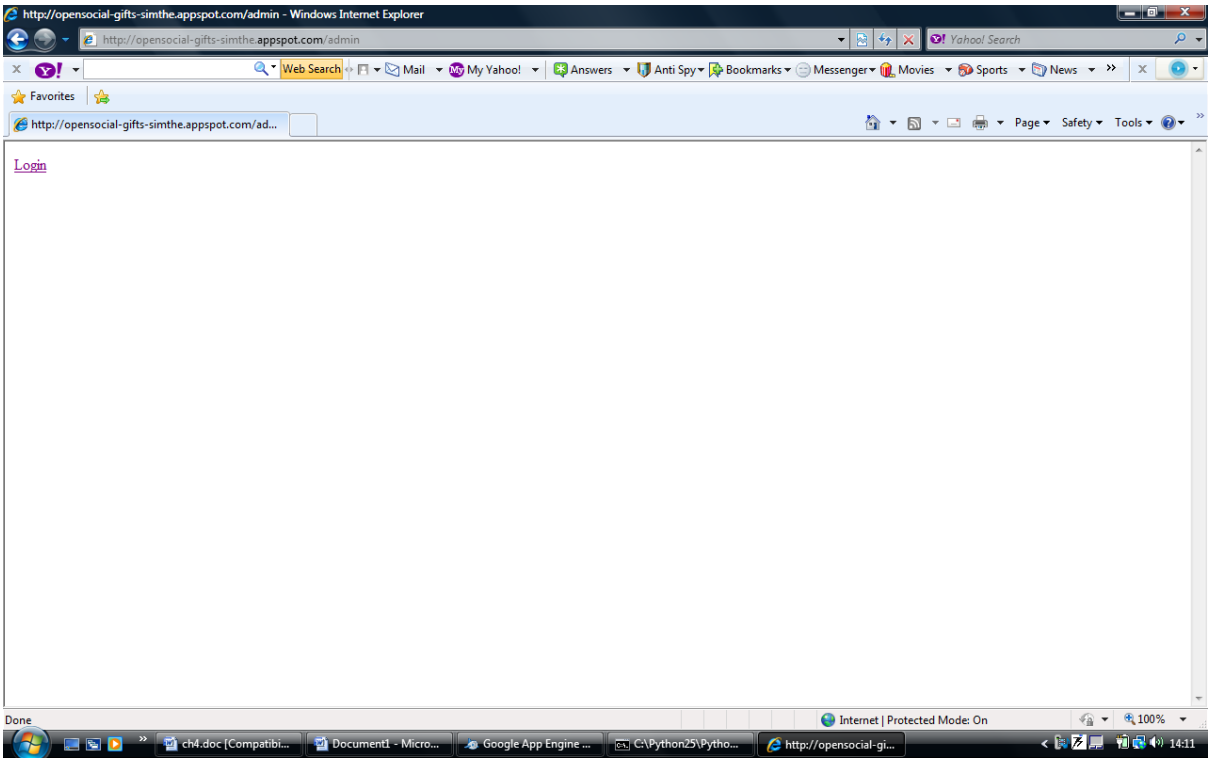

Figure. 5.1: Login page

We need to click on the “Login” link for further access.

We login to browse through the admin module of our application through our Google account.

We get the screen (as shown in Figure 5.3) displayed if we enter the valid username and password.

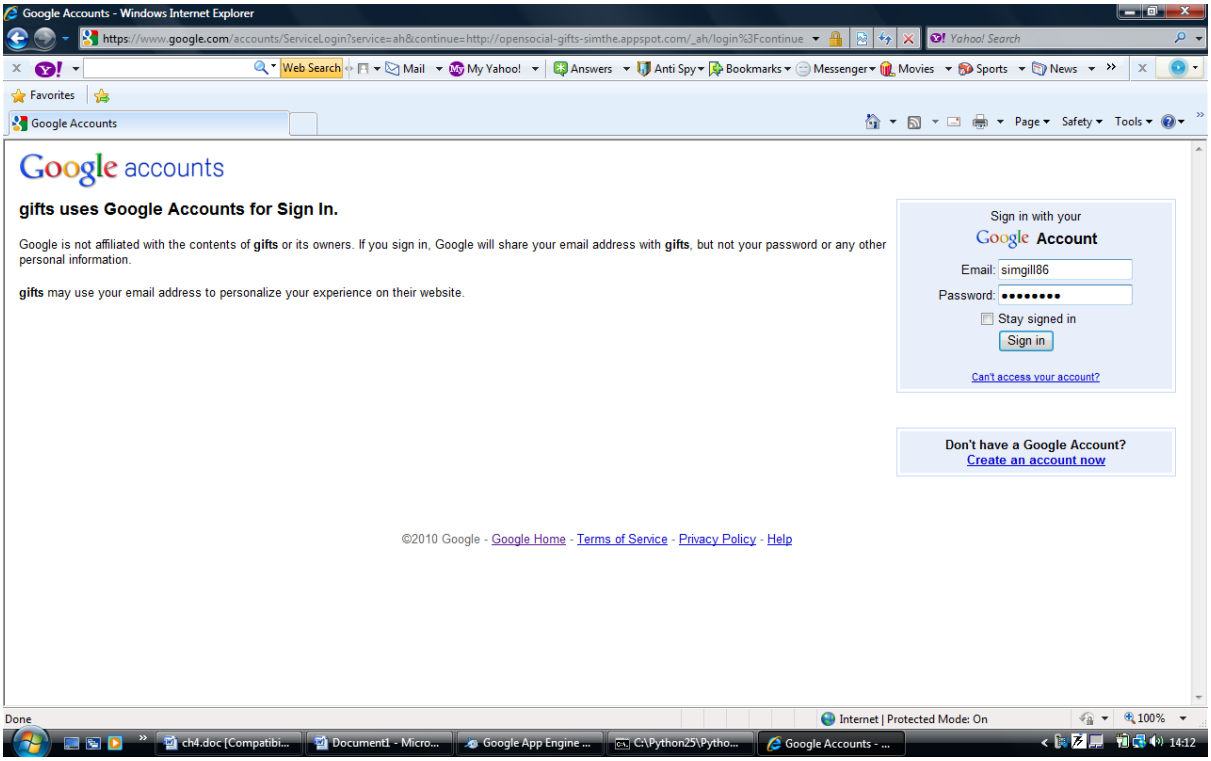

Figure. 5.2: Login Page for signing in through Google account

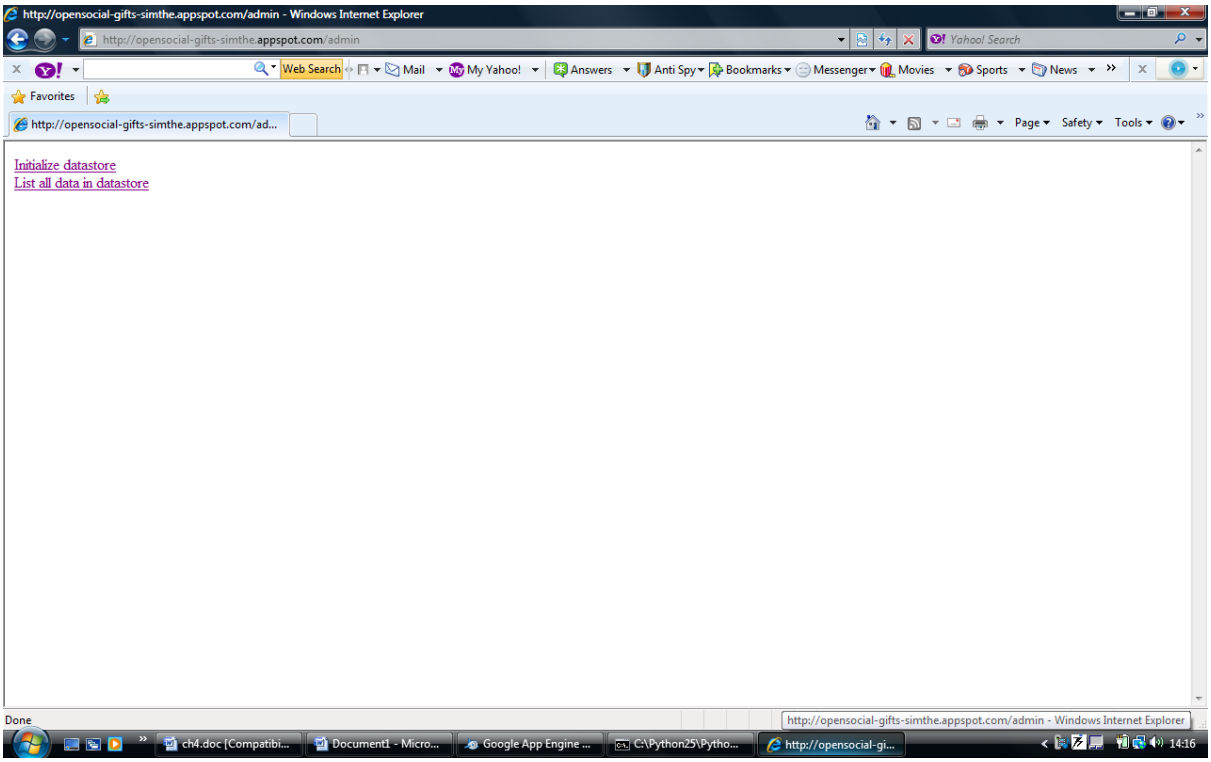

Figure. 5.3: Admin module home page

When we click on the “Initialize datastore” link, gifts and gift transactions are initialized at the backend.

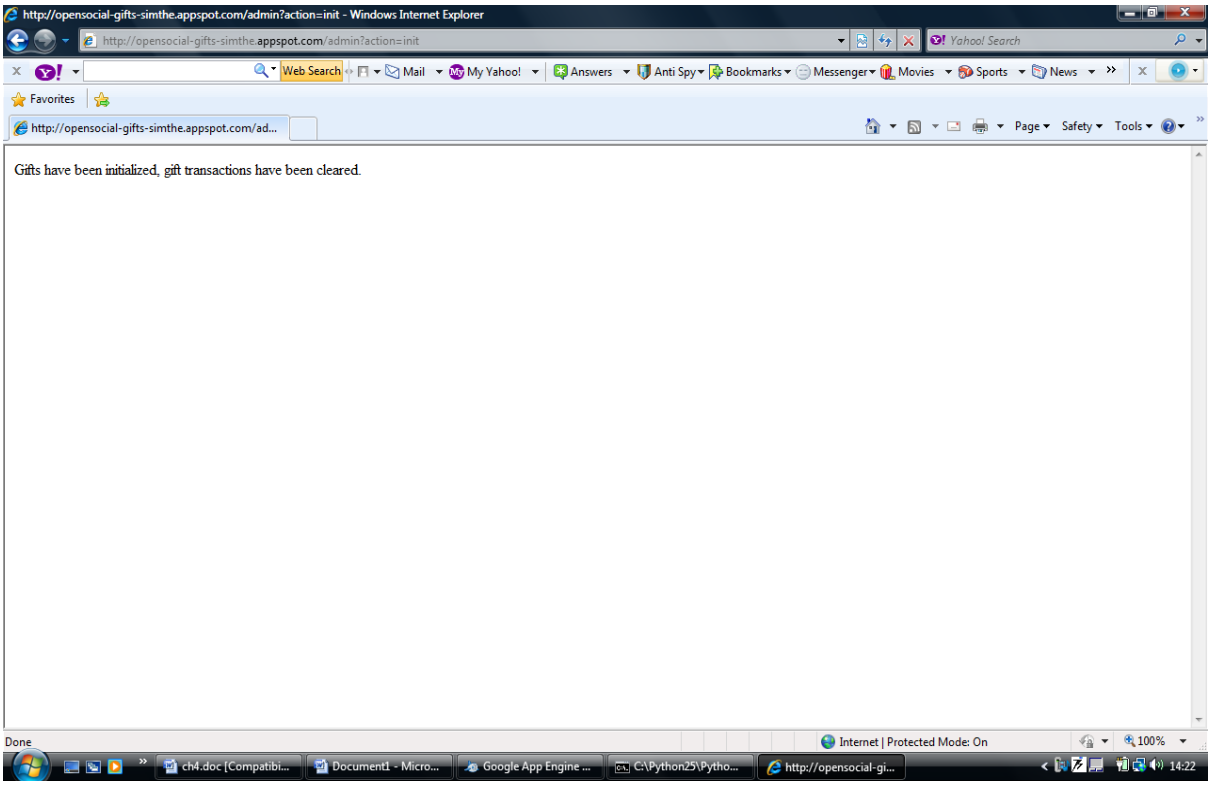

Figure. 5.4:Initialization of datastore

When we click on the “List all data in datastore “ link, the data from the datastore is displayed.

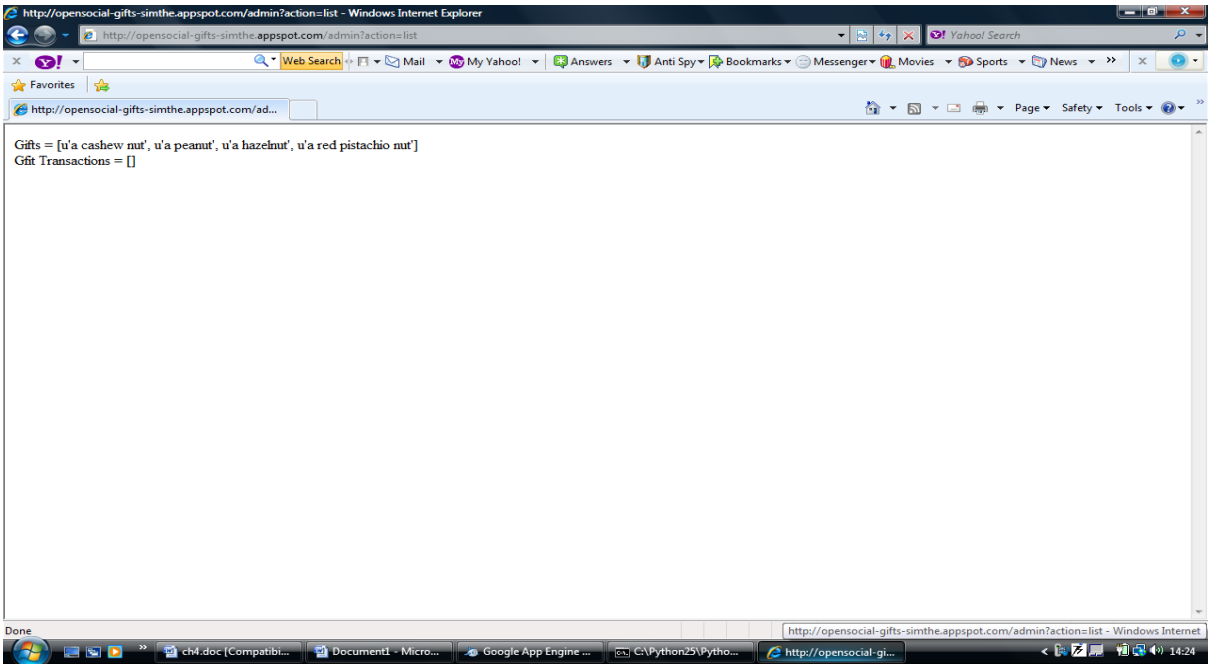

Fig. 5.5: Data retrieved from datastore

5.2 Browsing the gifts page

When we browse the url http://opensocial-gifts-simthe.appspot.com/gifts/, the gifts stored in the datastore are displayed.

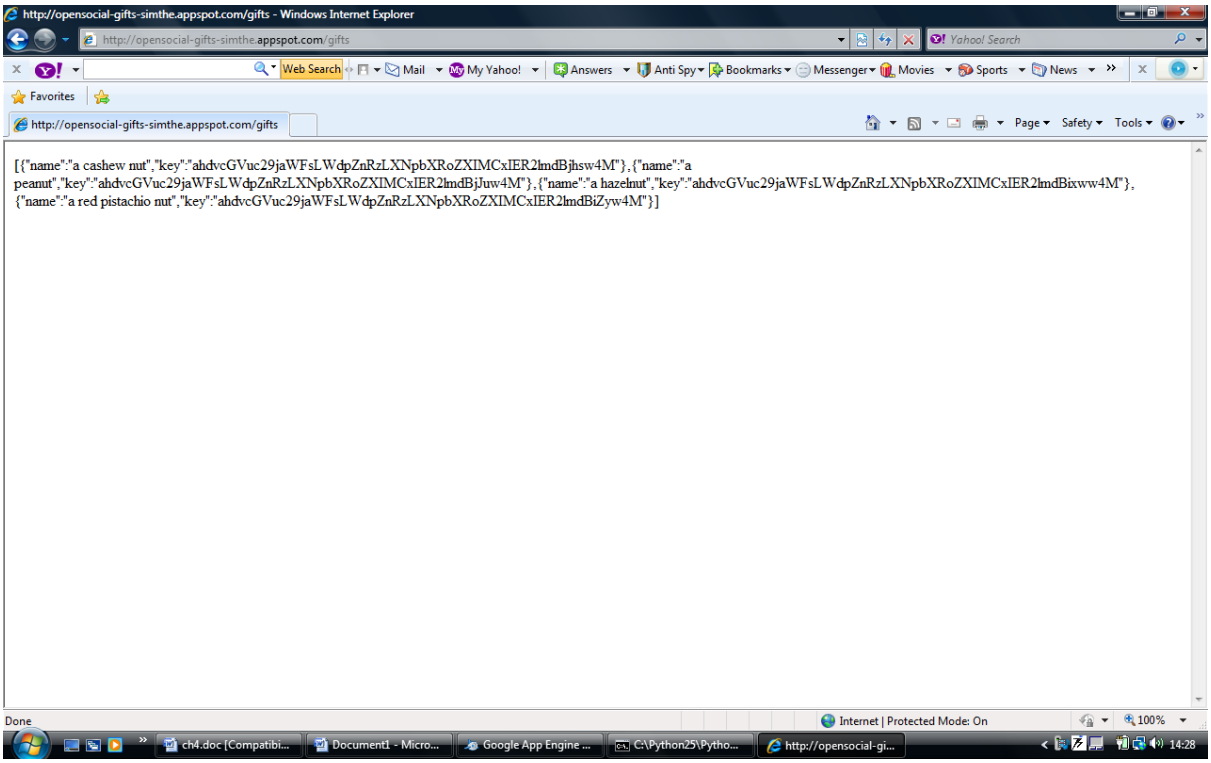

Figure 5.6: list of gifts retrieved from datastore

5.3 Browsing the application specification page

The application spec is displayed when we browse the url http://opensocial-gifts simthe.appspot.com/static/gifts.xml/

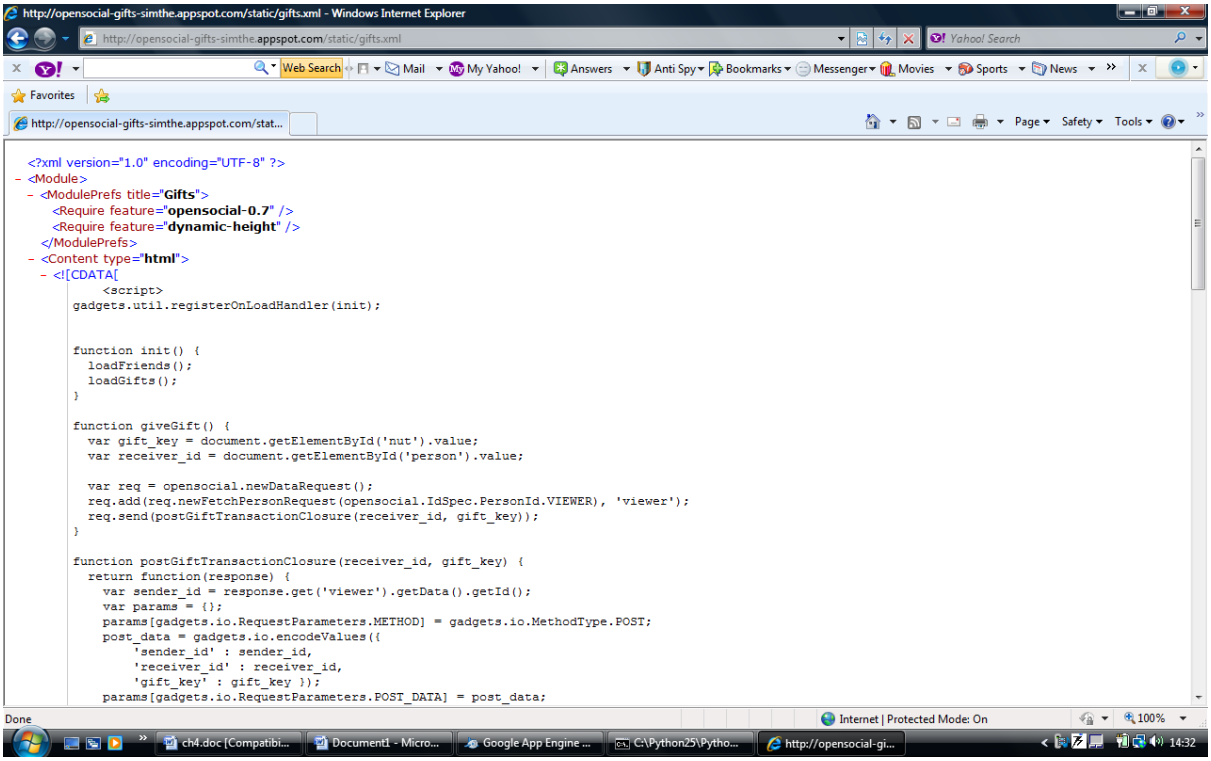

Figure 5.7: Application specification

5.4 Browsing the application installed in orkut sandbox

When we login into our orkut sandbox account, and visit the "Gifts" (home page of our application in orkut sandbox), we get the screen as shown in Figure 5.8.

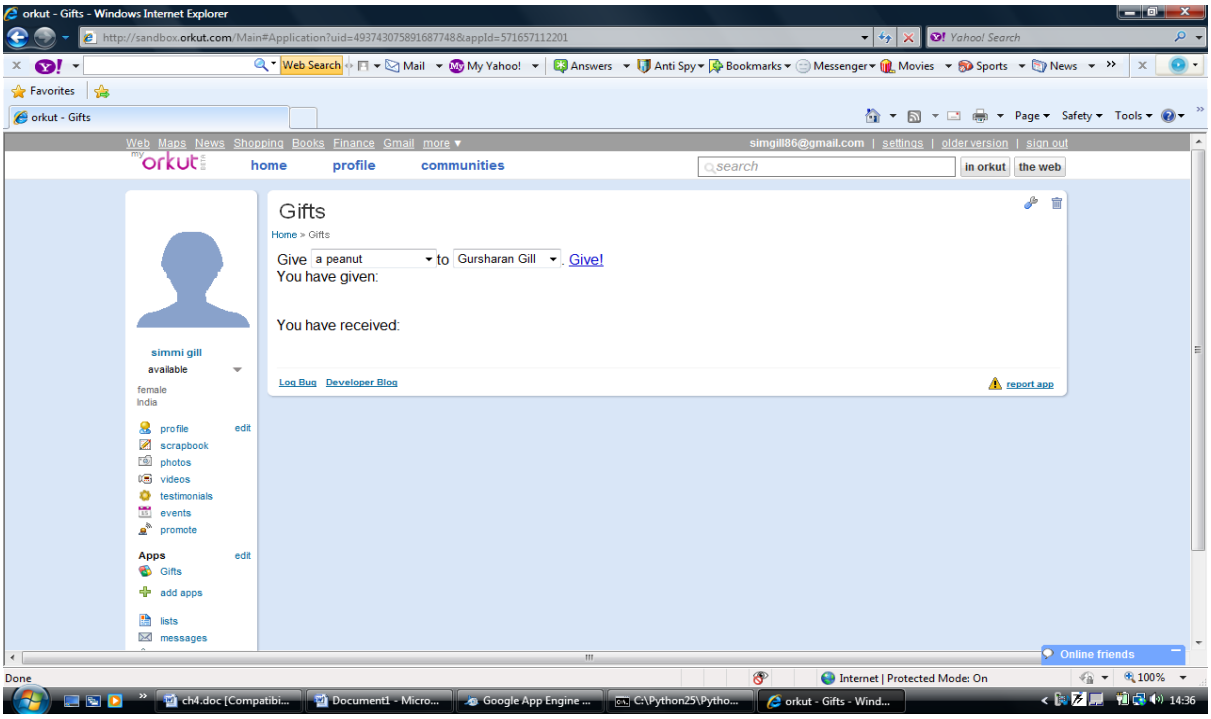

Figure 5.8: Home page of "Gifts" application in orkut

We have the list of gifts in one drop down menu and the list of friends in another. We can select one gift and one friend, and on clicking the “Login” link, the transaction will be displayed and saved in datastore.

Chapter 6

Conclusions and Future Scope

This chapter discusses about and the conclusions and the future scope of our thesis.

6.1 Conclusions

a) By developing applications on cloud (Google App Engine), we don't have to bother about setting up any servers.

b) The cloud vendor (Google) takes care of all the data storage.

c) The user doesn't has to bother about data security. Google takes care of everything.

d) Maintenance of servers is not all user's concern.

e) Google App Engine maintains all logs of our application.

f) Google App Engine keeps a track of the billing details of all applications.

g) Google App Engine provides some free quota for each application, beyond which the developers have to pay.

6.2 Summary of Contributions

a) By developing an application on Google App Engine, we have demonstrated the use of cloud.

b) Application development on Google App Engine shows how a cloud works.

c) Application has been deployed on Google App Engine with the url http://opensocial-gifts-simthe.appspot.com/

d) Application has been installed on Orkut sandbox.

e) Application can be shared among all friends on Orkut.

6.3 Future Research

a) Cloud is a new technology. It can be explored further by developing more complex applications.

b) Applications can be developed to explore areas like cloud security.

c) Applications can be developed to explore cloud scheduling.

d) Users can even set up their own private cloud environment by using software like Hadoop.

References

[1] http://www.infoworld.com/d/cloud-computing/what-cloud-computing-really-means-031

[2] http://knowledge.wpcarey.asu.edu/article.cfm?articleid=1614

[3] Jeffrey Voas, Jia Zhang. “Cloud Computing, New Wine or Just a New Bottle? “ . IT Pro, March/April.2009.

[4] Anthony T.Velte, Joby J.Velte, Robert Elsenpeter. “Cloud Computing”. Tata McGraw-Hill.

[5] Francesco Maria Agmerich, Gianni Fenu, Simone Surcis. “An Approach to Cloud Computing Network”.2008.

[6] Robert W. Lucky. “Cloud Computing”, IEEE Spectrum, May.2009

[7] http://www.kk.org/thetechnium/archives/2007/11/a_cloudbook_for.php

[8] Alexander Lenk, Markus Klems, Jens Nimis, Stefan Tai, Thomas Sandholm. “What’s Inside the Cloud? An Architectural Map of the Cloud Landscape”, Vancouver, Canada, CLOUD’09, May.2009, ICSE Workshop, 2009.

[9]http://www.informationweek.com/news/services/business/showArticle.jhtml?articleID=209904474

[10] Liang-Jie Zhang and Qun Zhou, “CCOA: Cloud Computing Open Architecture”. IEEE Computing society, 2009.

[11] Luis Ferreira, Viktors Bersti, Jonathan Armstrong, Mike Kendzierski, AndreasNeukoetter, MasanobuTakagi, Richard Bing-Wo, Adeeb Amir, Ryo Murakawa, Olegario Hernandez, James Magowan, Norbert Bieberstein”.IBM Red Books

[12] Ian Foster, Yong Zhao, Ioan Raicu, Shiyong Lu. “Cloud Computing and Grid Computing 360-Degree Compared”, 2009.

[13] Borje Ohlman, Anders, Eriksson, Rene Rembarz. “What Networking of Information Can Do for Cloud Computing”. IEEE computer society, 2009.

List of Publications

1.Simarpreet Kaur Gill, Dr.(Mrs.)Seema Bawa, " Application Development on Cloud Environment" communicated in journal of computing.